Protective Design - Mandatory Center of Expertise Technical Report 92-4

Facility And Component Explosive Damage Assessment Program

(FACEDAP)

Programmer's Manual

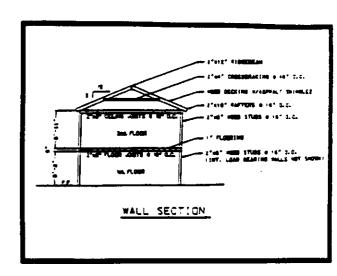
Version 1.2

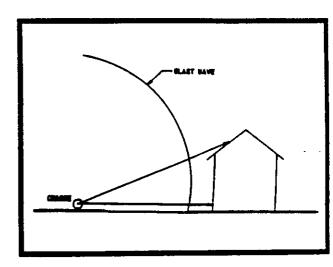
SwRI Project No. 06-5145-001 Contract No. DACA 45-91-D-0019

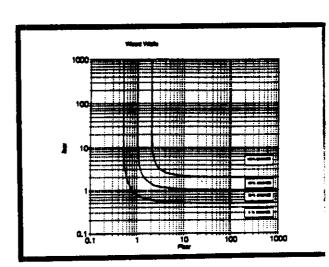
Prepared for:

Department of the Army Corps of Engineers, Omaha District CEMRO-ED-ST 215 N. 17th Street Omaha, Nebraska 68102-4978

Modified May 1994







Protective Design - Mandatory Center of Expertise Technical Report 92-4

Facility And Component Explosive
Damage Assessment Program

(FACEDAP)

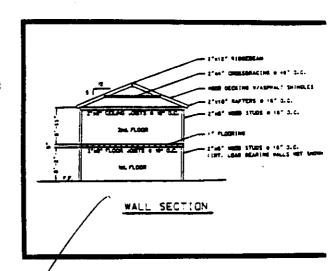
Programmer's Manual

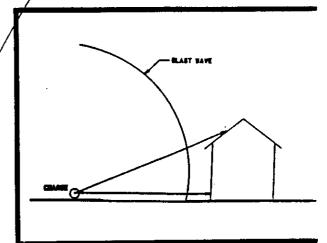
SwRI Project No. 06-5145-001 \
Contract No. DACA 45-91-D-0019

Prepared for:

Department of the Army
Corps of Engineers, Omaha District
CEMRO-ED-ST
215 N. 17th Street
Omaha, Nebraska 68102-4978

April 1993





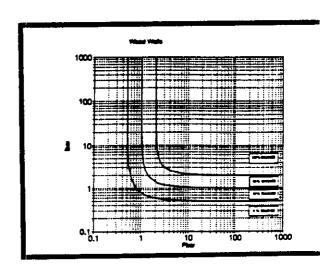


Table of Contents

Page
st of Figures iii
st of Tables iii
0 Introduction 1
0 Flow Diagrams for the Primary Executables in the FACEDAP Code
0 Software Used to Program the FACEDAP Code 4
0 Modifying the Problem Size Which Can be Analyzed with the FACEDAP Code
0 Compiling and Linking the FACEDAP Code
eferences
ppendix A: Indented Calling Trees for the FACEDAP Program
 -1: Indented Calling Tree for the FACEDAP Driver -2: Indented Calling Tree for BDAMPREP.EXE in the FACEDAP Preprocess Module -3: Indented Calling Tree for VALIDFIL.EXE in the FACEDAP Module -4: Indented Calling Tree for MAKEBDMA.EXE in the FACEDAP Analysis Module -5: Indented Calling Tree for BDAMA.EXE in the FACEDAP Analysis Module -6: Indented Calling Tree for BDAMPOST.EXE in the FACEDAP Postprocess Module
ppendix B: Description of Primary Subroutines Called in the FACEDAP Code

Table of Contents (Continued)

List of Figures

Page

Figure

1	Flow Diagram for FACEDAP Program Driver	2
	List of Tables	
Table		Page
1	Files Containing Dimensions Which Control Program Size	5
2	Major Arrays and Matrices in the FACEDAP Code	7
3	Matrices in BDAMA.EXE with Dimensions that Control Program Size	9
4	Information on Linking Major FACEDAP Executables and Output Files Created During Program Execution	11

1.0 Introduction

The purpose of this manual is to familiarize the reader with the basic architecture of the FACEDAP computer code. The information in the manual includes: 1) a detailed flow diagram and description of the primary programs and subroutines in the FACEDAP code; 2) a description of the software used by the code; 3) a description of the matrices and arrays which have dimensions which control the maximum problem size that can be run in the FACEDAP code; and, 4) instructions on compiling and linking the primary executables in the code. This manual is intended to be read in conjunction with the FACEDAP User's Manual [1], the FACEDAP Theory Manual [2], the IOSUB User's Manual [3], and the ESHELL User's Manual [4]. The FACEDAP User's Manual [1] describes use of the FACEDAP program and includes; one worked example problem, thirteen example buildings with example input files that are included on the program disk, and a listing of all error messages with explanation. The FACEDAP Theory Manual [2] discusses and assesses the theoretical approach used to determine building vulnerability to blast damage in the FACEDAP program. The IOSUB User's Manual [3] and ESHELL User's Manual [4] describe subroutines in these software packages which are used to control screen display in the FACEDAP code and run the FACEDAP driver using a minimum amount of computer memory.

The FACEDAP program is an approximate procedure for determining the vulnerability of common structural components and industrial buildings to explosive threats. It is not intended for use in blast resistant design or in any other situations where high accuracy is required. If the single component analysis option is invoked, the program calculates the blast damage to an input component from an input explosive threat using a procedure which is based on available experimental data and basic dynamic structural response theory. If the building analysis option is invoked, the building vulnerability, in terms of the percentage of building damage caused by the explosive threat, is calculated by the program in a two-step procedure. In the first step, the damage to each component in the building is calculated. In the second step, the calculated damage to each component is summed and divided by a value corresponding to total building failure to determine the percentage of building damage. Building reusability and replacement factors and the level of protection provided to assets within the building are also calculated. The theoretical approach used by the FACEDAP program is assessed and discussed in detail in the FACEDAP Theory Manual [2]. As the FACEDAP User's Manual [1] explains, input of the relatively large amount of required building information is facilitated by the FACEDAP program preprocessor, which automatically generates many building components, only requires input of unique building component properties sets, and automatically generates "dependencies" that define how building components support, and are supported by, other components.

The FACEDAP computer code consists of a driver, FACEDAP.EXE, three primary processors; the Preprocessor, the Analysis program, the Postprocessor; and a minor executable and several subroutines which set default configuration values and retrieve existing input files. These processors are illustrated in Figure 1. The executables in each processor shown in Figure 1 are "child programs" which are run, one at a time, as designated by input from the user into the driver program. In general, each child program reads a file on the hard disk with required input information, performs various calculations using the input information and additional information input by the user, displays some calculated information on the screen, and writes an output file which is used as input by one of the other child programs. The FACEDAP driver also allows the user to retrieve

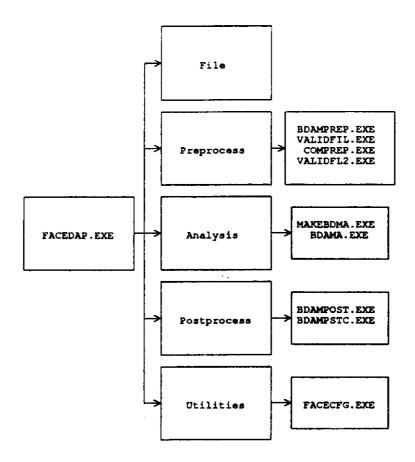


Figure 1. Flow Diagram for FACEDAP Program Driver

previous input files, establish program default directories, default printers, default print communications ports, set default screen colors, and shell out to DOS. These capabilities are included in the boxes marked "utilities" and "files" in Figure 1.

The FACEDAP Preprocessor reads all information defining the explosive charge weight and location and the components exposed to blast loading. In the case of a building analysis, the Preprocessor uses the fact that most buildings are comprised of a relatively small number of "unique" building components, which are used repetitively throughout the building construction, to reduce the effort required to input the required information. The user must initially break the building into large, planar wall/roof areas which are defined by four corner coordinates. All subsequent building input and output is initially defined in terms of one of the wall/roof areas and then in terms of components within that wall/roof area. This allows the user to more easily "manage" the relatively large amounts of detailed building information which must be input without the aid of a graphical display. It also allows the coordinates of building component endpoints to be input in terms of a local two-dimensional coordinate system in each wall/roof area rather than in terms of the global, three-dimensional coordinate system.

There are many checks built into the Preprocessor which issue warnings to the user during input if there appears to be an error in the input. The Preprocessor also makes extensive use of the IOSUB software, which has pre-coded subroutines that display user friendly form-type and spreadsheet-type screens to accept input. Finally, the Preprocessor includes a validation program which checks the user input and provides a specific error message for each error that is found. The FACEDAP program input can be simplified further in the future by the addition of a graphics package, which displays the spatial location and properties of input components. The availability of a graphics package would also enhance the program output.

If no errors are found by the validation program, the user can run the BDAMA.EXE program in the block labeled "Analysis" in Figure 1. This executable calculates building damage using the two-step procedure explained above. After the analysis has been completed, the Postprocessor displays building or single component blast damage information and blast load information. The Postprocessor makes extensive use of the IOSUB software to display blast damage information on each building component from a building analysis on spreadsheet-type screens in terms of the user-defined wall/roof areas of the building.

2.0 Flow Diagrams for the Primary Executables in the FACEDAP Code

Appendix A shows Calling Tree for each of the primary executables in the FACEDAP Code. Appendix A-1 shows the Calling Tree for the FACEDAP driver. Appendix A-2 shows the Calling Tree for BDAMPREP, the main Building Analysis Preprocessor executable. Appendix A-3 shows the Calling Tree for VALIDFIL, the primary Building Analysis error checking executable. Both BDAMPREP and VALIDFIL are part of the Preprocess module as shown in Figure 1. Appendix A-4 shows the Calling Tree for MAKEBDMA, which uses the information input into the Building and Single Component Preprocessors to write an input file for the BDAMA executable. Appendix A-5 shows the Calling Tree for BDAMA, which analyzes the building damage caused by the input explosive charge. Appendix A-6 shows the Calling Tree for the Building Analysis Postprocessor, BDAMPOST, which displays the calculated results. Appendices A-7 through A-9 show the Calling Trees for the Single Component Analysis executables. Appendix

A-7 is the Calling Tree for COMPPREP, which is the Single Component Analysis Preprocessor. The Single Component Analysis Validation routine, VALIDFL2, Calling Tree is shown in A-8. Appendix A-9 shows the Calling Tree for the Single Component Analysis Postprocessor, BDAMPSTC, which displays the component damage results.

The Calling Tree for the FACEDAP driver, in Appendix A-1, will be used as an example to illustrate how the Calling Trees in Appendix A and the subroutine purpose statements in Appendix B describe the flow of the FACEDAP program. The main program in the FACEDAP driver, which is also called FACEDAP, is the first program shown in the Calling Tree. This program calls the eleven indented subroutines shown below it. The order of these eleven subroutines is alphabetical, not the order in which the subroutines are invoked. All subroutines in the executables which call other subroutines follow the main program in the Calling Tree in alphabetical order. In the FACEDAP driver, two subroutines, namely CHGDIR1 and CHILD1, call other subroutines and therefore they are shown in alphabetical order under the main program with the (indented) subroutines they call. The calling arguments and the purpose of the subroutines shown in all the Calling Trees in Appendix A are listed and described in alphabetical order in Appendix B except as noted in the next two (2) paragraphs. The reader can understand the function CHGDIR1 by looking up this subroutine in Appendix B and reading the purpose statement. This standard method of presenting a program flow diagram is well suited for complex programs like the FACEDAP code which involve many subroutine calls.

Subroutines in the Calling Trees which are taken from the IOSUB software package without modification are described in the IOSUB User's Manual [3]. Only IOSUB subroutines which are used for overlay purposes, or are modified versions of the original IOSUB subroutine, are included in Appendix B. Filenames of modified IOSUB routines have a "B" proceeding the filename, such as BSPREAD.FOR instead of SPREAD.FOR. The subroutine call statement and the subroutine name within these files are not changed. The Calling Trees in Appendix A where generated by a useful IOSUB program called MKTRE. More information on this routine may be found in the IOSUB User's Manual.

Appendix C contains the routines which were added when the Single Component Analysis was added. Refer to Appendix C when an "*" appears following the subroutine/function name in the Calling Tree.

3.0 Software Used to Program the FACEDAP Code

The FACEDAP code is written in Microsoft Fortran, Version 5.1. As mentioned previously, many of the subroutines in the IOSUB software package are called by the FACEDAP code. These pre-coded subroutines, which are also written in Microsoft Fortran, control the most of the input and output in the code. Compiled versions of these subroutines are in the IOSUB libraries included on the FACEDAP Source Disk. A complete description of the IOSUB subroutines is included in Reference 3. Subroutines in the ESHELL software, which contain logic that minimizes the amount of memory used by the FACEDAP driver program, are linked with the driver program. This software enables all but approximately 3K bytes of the Driver program, FACEDAP.EXE, to be unloaded from memory while each child program is running, freeing more memory for the child programs.

None of the software in the ESHELL package is referenced in the Calling Tree in Appendix A-1. The ESHELL software, which is marketed by Kandu, Inc., is discussed in Reference 4. The IOSUB software is marketed by Business Systems Integration, in San Antonio, Texas.

4.0 Modifying the Problem Size Which Can be Analyzed with the FACEDAP Code

The FACEDAP Source Disk contains all program source files written or modified during this project, the ESHELL and IOSUB libraries which are linked with executables in the FACEDAP program, and batch files which contain the Microsoft Fortran compile and link commands necessary to build the program executables. Therefore, this disk contains all the code a user needs to modify the FACEDAP code and rebuild it. This section and the following sections contain programming information which helps explain the framework of the code. Most of the major matrices and arrays in the program are dimensioned using variables which are only defined once in an "include" file with the extension .INC. The only exceptions are the matrices in the BDAMA.EXE program, which are dimensioned in each subroutine where they are used as discussed at the end of this section. This makes it possible to change the size of arrays and matrices, and therefore the size of problem, by just changing the variables controlling the dimensions of the matrix in the appropriate include file. Table 1 shows the names of variables which define the dimensions of the major matrices and arrays, explains their use and their current values, and shows the files where they are located. Items 3, 5, and 6, in Table 1 refer to numbers in the SPRDHEAD.DAT and/or SPRDHED2.DAT data files which control the size of spreadsheets with input/output information that are displayed on the screen. These numbers are located in the header lines of the given spreadsheets in the SPRDHEAD.DAT and SPRDHED2 files. Spreadsheets are defined in this file by their numerical order in the file. NOTE: SPRDHEAD.DAT is used by the Building Analysis executable. SPRDHED2.DAT is used by the Single Component Analysis executable. The difference in these files is that the Single Component Analysis does not require Weighting Factors. Instead, a Span Length is required. In the Single Component Analysis the Span Length replaced the Weighting Factor unless the Span Length was already present, in which case the Weighting Factor was just deleted.

Table 2 shows the major matrices and arrays in the FACEDAP Code. It also shows the parameters in Table 1 which control the dimensions of the matrices, the variable type which is stored, and the purpose of each array or matrix. Major matrices in the BDAMA.EXE executable are shown separately in Table 3. The values of variables defining the maximum array and matrix dimensions in Table 2 have been carefully chosen so that most typical buildings can be input without exceeding the limits of the code. They have also been chosen to both limit the memory required by each separate child program so that the computer operating system and other typical background programs can run at the same time as the FACEDAP code, and allow each child program to maintain all required storage information in memory.

In spite of the efforts described above, it may still be necessary to increase the size of the maximum problem which can be analyzed with the FACEDAP code. The matrix dimensions which are considered most likely to require modification are those controlling the maximum number of components that can be input into the FACEDAP program in a single run and the sizes of the spreadsheet-type input/output screens. Item 1 in Table 1 shows that the FACEDAP code is currently set up so that it will analyze a maximum of 370 building components. This maximum causes the array, COMPG, which stores the building component information, to be very near the 64,000 byte limit for a matrix which is maintained in a single block of computer memory. As Tables 1 and 2

Table 1. Files Containing Dimensions Which Control Program Size

ITEM NO.	VARIABLE	FILE	VALUE	DESCRIPTION
1	max_num_components	BDAM.INC	370	Total components
2	max_wall_areas	BDAM.INC	50	Wall/Roof areas
3	Number of Rows in Input Screens for Building Geometry (Spreadsheet 26,28-34)	SPRDHEAD.DAT	150	Components per wall area
4	max_xmp_row	XMP.INC	150	Unique sets of component properties (total)
5	Number of Rows in Input Screens for Component Properties (Spreadsheets 1-24)	SPRDHEAD.DAT SPRDHED2.DAT	100	Component property sets per component type
б	Number of Rows in Spreadsheet 27 & max_generated_pairs	SPRDHEAD.DAT & BDAM.INC	50	Number of generated components from a master component
7	max_depend_pairs & dimension statements in BDAMA	BDAM.INC & BDAMA subroutines (see Table 3)	200	Total dependency pairs
8	max_column_ever	SPREAD.INC	17	Maximum columns on any spreadsheet
9	max_row_ever	SPREAD.INC	150	Maximum rows on any spreadsheet
10	total_spread_sheets	SPREAD.INC	34	Maximum number of spreadsheets
11	max_bdama_components & dimension statements in BDAMA	POSTPROCESSOR SUBROUTINES & BDAMA Subroutines (see Table 3)	400	Maximum number of components as allowed by BDAMA

Table 2. Major Arrays and Matrices in the FACEDAP Code

ITEM NO.	MATRIX NAME	VARIABLE TYPE	ROW DIMENSION	COLUMN DIMENSION	FILE WITH DIMENSIONING PARAMETERS	PURPOSE OF MATRIX
1	хтр	char*10	max_xmp_row	max_xmp_col	XMP.INC	Storage array for component properties
2	wall_table	char*11	max_row_ever	max_column_ever	WALLD.INC	Storage and spreadsheet array for wall/roof area definitions
3	compg	char*10	max_num_components	max_column_ever	COMPG.INC	Storage array for component geometry and dependency data
4	spread_table	int	17	total_spread_sheets	SPREAD.INC	Contains option instructions for each column of each spreadsheets
5	work_table	char+11	max_row_ever	max_column_ever	SPREADT.INC	Work array used for displaying spreadsheets
6	load_buf	char*10	4	N/A	LOADBUFF.INC	Form array used for load definitions
7	problem_buf	char+70	4	N/A	PROBBUFF.INC	Form array for title and descriptions
8	damage_buf	char+8	4	N/A	BDAMPOST.FOR	Form array for building damage
9	clink	int	max_xmp_row	N/A	CLINK.INC	Linklist for the XMP array
10	mlink	int	max_num_components	N/A	LINKLIST.INC	Linklist for the COMPG array
11	comp_def_buf	char+31	3	N/A	COMPDEF.INC	Form array used for component selection in the Single Component Analysis Program

show, COMPG is a 370 x 17 matrix, where each stored field is 10 characters, so that 62,900 bytes of memory is required to store this matrix. The COMPG matrix can be modified to store more building components by creating a separate matrix which stores the last four columns in the COMPG array and creating a pointer which links the corresponding rows in COMPG and in the new matrix. Column 1-13 in the COMPG matrix store component geometry data and columns 14-17 store the component dependencies.

Spreadsheet-type input screens, which are simply referred to as spreadsheets, are used in the Building and Single Component Preprocessors and the Building Postprocessor for input and output of building/component information. Each spreadsheet is defined by a Spreadsheet Number which refers to its position in the SPRDHEAD.DAT and SPRDHED2.DAT files. These files define the number of rows, number of columns, number of header lines, text in the column headers, text in the help messages, defaults input values and several other pieces of information for each spreadsheet. All spreadsheets, with the exception of WALL_TABLE, are displayed by the BSPREAD.FOR subroutine using the matrix, WORK_TABLE, which can be referred to as a work area. This work area is equivalenced to other matrices. However, the maximum dimension of any spreadsheet is governed by the dimensions of WORK_TABLE.

Items 3, 5, 6, in Table 1 show the current number of rows for each spreadsheet in the SPRDHEAD.DAT and/or SPRDHED2.DAT files. The maximum number of rows in any spreadsheet can be increased up to a maximum of 150 rows by changing the number of rows in the header line of the given spreadsheet in the SPRDHEAD.DAT and SPRDHED2 files. The limit for the maximum number of rows in a spreadsheet is set with the parameter max_row_ever in Table 1. The number of columns displayed to the screen for any given spreadsheet can be increased up to the limit of 17 columns using the steps shown below. The limit on the maximum number of columns in a spreadsheet is set with the parameter max_column_ever in Table 1.

- 1) Update SPRDHEAD.DAT and SPRDHED2.DAT Change the number of columns in the header line of the given spreadsheet in the SPRDHEAD.DAT and SPRDHED2.DAT file. See the purpose statement for subroutine GETSPRED in Appendix B for an explanation of how the various spreadsheets and header lines are stored in these files.
- 2) Update OPTIONTB.DAT and OPTION2.DAT These files contain information controlling special options that apply to each column in each spreadsheet. Each line in this file controls options for the spreadsheet in the SPRDHEAD.DAT and SPRDHED2.DAT files. The option control parameters on the second line, for example, control the second spreadsheet called by the program. Update the option control parameters for each new spreadsheet column as explained in the comment statements at the top of the BDAMPREP.FOR and BDAMPOST.FOR files. OPTIONTB.DAT is used for the Building Analysis and OPTION2.DAT is used for the Single Component Analysis.

- 3) Update GETVARS.FOR and GETVARS2.FOR Column numbers used for default calculations in spreadsheets must be updated in GETVARS.FOR for the Building Analysis spreadsheets and GETVARS2.FOR for the Single Component Analysis spreadsheets.
- 4) <u>Update CALCDEFS.FOR</u> Formulas used to calculate default component properties in the first 24 spreadsheets in SPRDHEAD.DAT and SPRDHED2.DAT must be updated with new variable numbers as per assignment in GETVARS.FOR and GETVARS2.DAT.
- 5) Update CONV4PRT.FOR Update EXCLUDE_COL and MAX_COMP_COL arrays. EXCLUDE_COL is used to determine columns on the spreadsheet that are character types and therefore do not need conversion from character to real or integer. This routine is used to print the Preprocessor reports. NOTE: The data statement contains the excludes for the Building Analysis spreadsheets. These excludes are modified using assignment statements when this routine is used for a Single Component Analysis spreadsheets.
- MAX_COMP_COL matrices. EXCLUDE_COL is used to determine which columns on the spreadsheet are excluded from the check for a 0 value. MAX_COMP_COL contains the maximum number of columns displayed on each spreadsheet and is used for looping control when checking columns with EXCLUDE_COL. NOTE: The data statement contains the excludes for the Building Analysis spreadsheets. These excludes are modified using assignment statements when this routine is used for a Single Component Analysis spreadsheets.
- 7) Re-link BDAMPREP using LXBDAMP. Re-link COMPPREP using LXPREP2.
- 8) Re-link VALIDFIL using LXVALID. Re-link VALIDFL2 using LXVALID2.
- 9) <u>Update BDAMAWRLFOR and subroutines in the BDAMA</u> executable if the information read into the new columns affects these subroutines.

The ramifications of increasing the number of columns in a given spreadsheet above the current 17 column limit are much more extensive. All the storage matrices, XMP, COMPG, and WORK_TABLE, must be increased as well as several data files and subroutines. As previously pointed out, COMPG is already extremely close to the maximum size which can be contained in a single block of memory so it is not advisable to take this step with out making the suggested storage changes in COMPG.

The BDAMA.EXE executable, which is shown in the Analysis block in Figure 1, was written during a previous project and does not pass any information using common blocks. The sponsor of this work precluded the use of common blocks and therefore matrices and arrays are dimensioned separately in each subroutine where they are used. Modifications were made during the development of the FACEDAP code but they did not affect the structure of this executable. Table 3 shows the major matrices in this executable, the subroutines where they are dimensioned, and their current values.

Table 3. Matrices in BDAMA.EXE with Dimensions that Control Program Size

ITEM NO.	MATRIX	SUBROUTINE	DIMENSION
1	dam	COMPD.FOR COMPR.FOR OUTPOST.FOR SUM.FOR BDAM1.FOR	(400,8)
2	newdep	DEPMOD.FOR	(400,2)
3	idep	DEPMOD.FOR BDAM1.FOR DEPIN.FOR READAT.FOR	(200,2)

5.0 Compiling and Linking the FACEDAP Code

The subroutines shown in the Calling Trees in Appendix A control the input, output, and calculation scheme of the FACEDAP code. These files are all included on the FACEDAP Source Disk with root names which consist of the subroutine names shown in the Calling Trees followed by the extension .FOR. The Source Disk also includes files with the extensions .BAT, .LNK, and .INC, and .DAT, which are needed in order to compile and link the FACEDAP code. All the files required to build any of the FACEDAP code executables are in the archived file FACECODE.ZIP on the source disk. This disk includes a README.DOC file with instructions on how to unarchive the individual files from FACECODE.ZIP using the PKUNZIP.EXE file which is also on the source disk.

The .INC files contain common blocks that pass variables, arrays, and matrices between subroutines in the code (except in BDAMA.EXE where there are no common blocks). These files, which are called "include" files, are referenced with the fortran INCLUDE statement in all subroutines where the common blocks in the files are used. Information required by the FACEDAP code to set up input and output screens is programmed into files with the extension .DAT. The

.DAT files are ASCII character files which contain headings and help messages displayed by Preprocessor and Postprocessor screens as well as control parameters which define the size of each spreadsheet display.

The .BAT files contain the Microsoft Fortran compile and link commands necessary to build the executables in the FACEDAP code. During the development stage of the program, compilation was done using the FPC.BAT file provided on the Source Disk. This batch file contains the following two statements:

fl /FPi /Fs /Zi /Ge /Gt70 /c /4ccd /4Yb /Od /4Yd /G2 %1.for

rem fl /FPi /Fs /Zi /Ge /Gt70 /c /Od /4Yd /G2 %1.for

The first line contains the compilation statement for what is considered the full debugging mode. This statement causes extended error handling and bounds checking to be performed. The second line is a partial debug. It does not perform any bounds checking or extended error handling. Both, however, disable the compiler optimization and require that all variables be declared. Both methods generate the symbolic information required by the Microsoft CodeView Debugger. The remark command, REM, can be switched to remark out the command which is not used. The link method used while in the debugging mode also enables CodeView. The following statement shows the link text used.

link/CO/SE:512

The size of the executable generated in these modes is much larger than that generated in the production mode. The compilation command for the production mode, which is shown below, is contained in FPCPRO, BAT on the Source Disk.

fl /FPi /Gt70 /G2 /c %1.for

This command invokes the compiler optimizer and removes all Code View requirements. The link used for the production mode packs the executable in order to reduce the size of the executable.

link/SE:512/EXEPACK

Table 4 shows the link batch files (LX*.BAT files) required to create a new version of each of the executables included in the FACEDAP code. This table also shows the files created and read by each executable during execution. Most of the files which are created during execution have a user designated root name (the name entered as the "save file" name) which is referred to Table 4 with an asterisk. The LX*.BAT files have both debug mode and production mode link statements in them. One of these statement should always be remarked out.

Most of these link batch files invoke a *.LNK file, which is the link response file. The asterisk refers to the name of the executable which is generated with the file. These link response files list all the subroutines and the IOSUB and ESHELL libraries that are required in the link command which creates the executable. Prior to running any of the link batch files, the user must compile all .FOR files required by the executable using Microsoft Fortran. The .INC files referenced in the subroutines that are linked, as well as IOSUB and ESHELL libraries on the FACEDAP Source Disk, must also be available in the directory where the executable is linked.

To facilitate having the required files for generating an executable, batch files were created to extract the required files from the FACECODE.ZIP source archive. By running the appropriate batch file, all FORTRAN, include, libraries and batch files necessary for the executable generation, will be extracted and placed in the current directory. This prevents having to have all source files present when only one executable is to be modified.

Table 4. Information on Linking Major FACEDAP Executables and Output Files
Created During Program Execution

EXECUTABLE FILE NAME	EXTRACT BATCH FILE	LINK BATCH FILE	INPUT FILES	OUTPUT FILES
FACEDAPEXE	XFACE.BAT	LXFACE.BAT	FACE.CFG MAINTITL.DAT *.BLG	N/A
BDAMPREP.EXE	XPREPB.BAT	LXBDAMP.BAT	FACE.CFG OPTIONTB.DAT SPRDHEAD.DAT PRBTITLE.FRM LOADDEF.FRM *.BLG	*.BLG *.REP
VALIDFIL.EXE	XVALIDB.BAT	LXVALID.BAT	FACE.CFG *.BLG	*.BLG *.ERR
MAKEBDMA.EXE	XMAKEB.BAT	LXMAKEB.BAT	FACE.CFG •.BLG	BDAMA.IN
BDAMA.EXE	XBDAMA.BAT	LXBDAMA.BAT	*.BLG BDAMA.IN	*.PST
BDAMPOST.EXE	XPOSTB.BAT	LXPOST2.BAT	FACE.CFG OPTIONTB.DAT SPRDHEAD.DAT BUILDAM2.FRM LOADPOST.FRM *.BLG *.PST	*.REP
COMPPREP EXE	XPREPC.BAT	LXPREP2.BAT	FACE.CFG OPTION2.DAT SPRDHED2.DAT PRBTITLE.FRM LOADCOMP.FRM COMPDEF.FRM COMP*BLG	COMP*.BLG COMP*.REP
VALIDFL2.EXE	XVALIDC.BAT	LXVALID2.BAT	FACE.CFG COMP*.BLG	COMP*.BLG COMP*.ERR
BDAMPSTC.EXE	XPOSTC.BAT	LXPOSTC.BAT	FACE.CFG BUILDAMC.FRM LOADCMPP.FRM COMP*.BLG COMP*.PST	COMP*.REP

^{*} Indicates User assigned problem name, which is assigned in the Preprocessor, BDAMPREP.EXE or COMPPREP.EXE.

- Oswald, C.J., and Skerhut, D.c. "FACEDAP User's manual," Contract No. DACA 45-91-D-0019, U.S. Army Corps of Engineers, Omaha District, April 1993.
- 2. Oswald, C.J., "FACEDAP Theory Manual," Contract No. DACA 45-91-D-0019, U.S. Army Corps of Engineers, Omaha District, April 1993.
- 3. IOSUB: A Library of Input and Output SUBroutines for use with Microsoft® Fortran, Version 2.0 User's Manual, Business Systems Integration, San Antonio, TX.
- 4. Empty Shell Version 1.1 Documentation, Kandu, Inc., Hamilton, NJ.

XV S

<u>X</u> .5A.

XV X

Appendix A Indented Calling Trees for the FACEDAP Program

Indented Calling Tree for the FACEDAP Driver

```
facedap
     chgdirl
     child
    clng
    cls
     curof
     curon
    dialog
    erabox
    fclose
     fdel
     fopen
     fsize
     getlun
     getmod
     hlpmsg
     idtmc
     imcoff
     inpfil
     isxlim
     isylim
     magbox
     popmen
     putstr
     rdcfg
     stdmsg
     wrcfg
chgdirl
     chdir
     clng
     dialog
     edtfld
     erabox
     getdir
     mkdir
     msgbox
child1
     clng
     cls
     espath
     getcur
     getmod
     hlpmsg
     idtmc
     imcoff
```

Indented Calling Tree for BDAMPREP.EXE in the FACEDAP Preprocess Module

charge clng winstr charge clng winstr charge clng winstr chgatr curon clng dialog hlpmag cls drawbo leftj comprop drawbo leftj curon cing curon curon erabox clng editform geting dialog editform erabox clng editform erabox hlpmag chgatag imcoff geting editform erabox hlpmag chgirl chdir geting geting putstr dialog geting geting editform geting editing putstr dialog geting editform repasc edifid hlpmag rightj erabox light erabox incomplete dialog geting editing strukin maker charge geting izchr spreadop geting mincoff winchr maybox clng geting isxim buildgeo clng hlpmag izghr isxim cdepend gixmyrow hlpmag prepprint cls clng hlpmag hlpmag prepprint cls erabox chk2way prepprint cls erabox chg winidex satmod hlpmag izchr winidex satmod hlpmag izchr leftj repasc erabox chwin putstr cuase left putstr compdef winidex satmod showne izchr left leftj repasc erabox chrwin putstr showne clng showne izchr clng showne clng drawbo strukin putstr chkmastr clng drawbo strukin putstr strong clng clng clng stronger clng clng clng clng stronger clng clng clng clng clng clng clng clng	bdamprep	chgatr	spread
charge chgatr chgdirl chgdirl clng dialog cls drawbo cls compprop curof drawhl compore curof drawhl compore curon dialog differ dialog dialog dialog dialog differ dialog dialog differ dialog differ dialog differ dialog differ dialog differ dialog differ dialog dialog differ dialog dialog differ dialog deditform dialog		chrwin	strwin
chgatr chgdirl clng clng clng dialog cls drawbo leftj compprop curof curof drawvl curon dialog getimp curof curon dialog getimp editform erabox filexist idtmc getarg getdata getiun magbox clng getiun magbox clng inpil iuxii buildgeo clng inpil iuxii buildgeo clng getimprow hlpmag preeprit cls leftj putstr compdef winindex preepas reset framdef getwname hlpmag i2chr ucase savedata getwname hlpmag i2chr izchr ucase leftj leftj leftj leftj leftj leftj leftj leftj stupechk delagen delagen delagen delagen delagen delagen delagen delagen delagen clng getimpos winindex clng drawbo strwin delagen dialog group putstr xyz2 compdef repasc colacdefs repasc reset chrwan clng drawbo strwin delagen delftj getip regasc chkmastr getomp getiun getip getiun getip getiun getip getiun getip getiun getip getiun getip getiun geti		clng	
chgdirl curon		curof	
clng dialog hipmag cls drawbo left; compprop drawhl repase curof drawhl repase curof drawhl charge curof drawhl charge curof drawhl charge curon erabox clng dialog getinp editform erabox hipmag hipmag hipmag filexist idtmc chgdirl chdir getarg imcoff chdir getarg imcoff chdir getarg imcoff chdir getarg imcoff chdir dialog getund repase detfild hipmag right; erabox i2chr spreadop getind midte strwin mkdir incoff winchr maghox charge isxlim buildgeo clng isylim capend grapac chemical hipmag right; erabox isxlim buildgeo clng hipmag repase cls left; putstr compdef winchr rdofg dialog xy22 repase erabox chkcomp repase erabox chkcomp showme i2chr ilchr ilchr incose savedata getwname hipmag i2chr clng showme i2chr itypech chcomp fucase left; left	_	curon	_
compprop curof drawwl charge curof drawwl charge curon gethox clng dialog gethod editform erabox hlpmag chigg editform speace erabox hlpmag chigg getdata msgbox clng getdata msgbox clng getwod repasc editform getmod repasc editform ilchr spreadop getdir idtmc strwin mkdir incoff winch msgbox inpfil winstr chk2way isxlim buildgeo clng isylim cdepend gtxmprow msgbox clng hlpmag prepprnt cls leftj putstr compdef wnindex rdoff dialog xy22 reset framdef clng sewadata getwname hlpmag setmod hlpmsg i2chr i2chr itypech clms sowme leftj leftj bselect msgbox rightj gtomprow chwin setmod hlpmag setmod hlpmag i2chr clng showme clng drawbo walldef dialog repasc erabox ckcomp chrwin spiss winstr chkmastr clng showme leftj leftj bselect msgbox rightj fucase walldef dialog strwin delgen ixpeck sort st clng clng drawbo walldef dialog strwin delgen fucase walldef dialog putstr chkmastr clng showme clng drawbo walldef dialog putstr clng repasc calcdefs chkmastr clng strwin delgen fucase walldef dialog strwin delgen fucase walldef dialog putstr clng clng getspred putstr clng clng strwin delgen clng drawbo getdpnd genit repasc clng getspred putstr repasc strwin putstr repasc clng getspred putstr repasc clng getspred putstr repasc strwin putstr repasc clng getspred putstr repasc strwin putstr strwin putstr strwin putstr strwin putst	-	. •	• . •
curof	cls		
curon dialog getinp editform elitform getmod erabox filexist idtmc chgdirl getarg imcoff chdir getatad msgbox clng getmod repasc edtfild hlpmsg rightj erabox ilchr spreadop getdir idtmc strwin mkdir imcoff winchr msgbox inpfil winstr chk2way isxlim buldgeo clng isylim cdepend gtxmprow hlpmsg prepprnt cls leftj putstr compdef winchr getmod hlpmsg repasc erabox chkcomp repasc erabox chkcomp reset framdef clng savedata getwname hlpmsg setmod hlpmsg ilchr ucase leftj leftj bselect msgbox rightj leftj bselect msgbox right chrwin clng drawbo strwin delgen dialog strwin drawbo winstr chkmastr clng drawbo winstr chkmastr clng drawbo winstr chkmastr clng drawbo winstr chmastr clng drawbo winstr clng drawbo winstr getcompgr putstr xy2 compdef repasc calcdefs chkmastr clng drawbo winstr getcompgr putstr xy2 compdef repasc calcdefs chkmastr clng drawbo winstr getcompgr putstr xy2 compdef repasc calcdefs chkmastr clng drawbo getwnod winstr getcompgr ismpos windex leftj stdnsg putstr xy2 compdef repasc calcdefs chkmastr clng chrwin gendpnd leftj stdnsg winchr clng getspred putstr repasc clng getspred stdnsg winchr cls gegroup betmod hlpmsg putstr cals gegroup putstr repasc cls getspred putstr winstr	compprop		
dialog getinp editform erabox hlpmag filexist idtme character filexist idtme character getarg imcoff chdir getarg imcoff chdir getarg imcoff chdir getarg getmod repasc editid putstr dialog getmod repasc editid hlpmag right) erabox illem mkdir ingoff idtme strwin mkdir imcoff winchr ingil winstr ch2way isxlim buildgeo clng isylim cdepend gtxmprow msgbox clng hlpmag repprot cls left; windex received a sexual getwame hlpmag illeft; chrwin chwaracter msgbox reset getwame hlpmag illeft; chrwin putstr dialog xyz2 ch8cmag repsch getwame illeft; left) left; chrwin putstr chymag illeft; left; left; chrwin putstr chymagox right; left; left; left; readke xyz2 compdef dialog getwame clng drawbo strwin delgen dialog getwame illeft; left; readke xyz2 compdef dialog getmod winstr getcompgr putstr xyz2 compdef chkmastr clng setwame illeft; left; readke xyz2 compdef chkmastr clng getwame illeft; left; readke xyz2 compdef chkmastr clng getwame illeft; readke xyz2 compdef chkmastr clng getwame clng drawbo gtompgr left; gen clng clng clng clng clng clng clng cln	curof	 	<u> </u>
editform getmod erabox hlpmsg hlpmsg filexist idtmc chdir getarg imcoff chdir getarg imcoff chdir getarg getdata msgbox clng getmod repasc edtfid hlpmsg rightj erabox i2chr spreadop gettim strwin mkdir msgbox idime imcoff winchr msgbox clng i2chr spreadop getdir idtmc strwin mkdir msgbox inpfil winstr ch82way inpfil winstr ch82way isxlim bulldgeo clng isylim cdepend gtxmprow hlpmsg prepprnt cls leftj leftj putstr compdef winindex xyz2 repasc erabox ch8comp reset framdef clng savedata getwname hlpmsg i2chr i2chr i2ybechk exemod hlpmsg i2chr i2ybechk chrwin getwname i2chr itypechk leftj le	curon		_
erabox hlpmsg hlpmsg filexist idtmc chdir getarg imcoff chdir getdarg imcoff chdir getdard magbox clng getlun putstr dialog getwod repasc edtfid right; getmc strwin mkdir msgbox i2chr idtmc strwin mkdir msgbox clng i2chr idtmc winchr msgbox clng hlpmsg chyslim cdepend gtxmprow hlpmsg prepprht cls leftj hlpmsg prepprht cls leftj winder savedata getwname hlpmsg i2chr leftj left			
filexist idtmc chgdirl getarg imcoff chdir getarg imcoff chdir getarg getdata msgbox clng getlun putstr dialog getmod repasc edtfld hlpmsg rightj erabox i2chr spreadop getdir mkdir msgbox inpfil winchr msgbox clng idtmc winchr msgbox clng clng isxlim buildgeo clng isylim cdepend gtxmprow hlpmsg repasc clng hlpmsg repasc erabox clng hlpmsg repert compdef winchr putstr compdef windex setmod hlpmsg i2chr i2chr itypechk leftj			
getary			
getdata msgbox clng getlun putstr dialog getmod repasc edtfld hlpmsg rightj erabox i2chr spreadop getdir idtmc strwin mkdir imcoff winchr msgbox inpfil winstr ch2way isxlim buildgeo chk2way isxlim cdepend gtxmprow msgbox clng hlpmsg prepprnt cls leftj prepprnt cls leftj putstr compdef wnindex redfg dialog xyz2 repasc erabox chkcomp reset getwname hlpmsg savedata getwname hlpmsg setmod hlpmsg i2chr savedata getwname hlpmsg setmod hlpmsg i2chr chrwin putstr chkmastr chrwin	 		
getlun putstr dialog getmod repasc edtfid getmod repasc edtfid ilzchr spreadop getdir ilzchr strwin mkdir imcoff winstr chk2way inpfil winstr chk2way clng dtmpmod dtmpmod isxilim buildgeo chk2way clng dtmpmod dtmpmod putstr compdef windex syz2 chkcomp chkcomp chrwin cheftj leftj clng chkmastr dtemped clng chkmastr dtemped getmo			
getmod	2		
Setup		-	_
i2chr spreadop getdir idtmc strwin mkdir mkdir mkdir msgbox inpfil winstr cheway inpfil winstr cheway isxlim buildgeo clng gtxmprow msgbox clng hlpmsg cls leftj mutstr compdef windex yzzz repasc erabox chemastr clng savedata getwname hlpmsg i2chr sowe leftj	_		
idtmc			getdir
imcoff winstr ch2way inptil isxlim buildgeo clng gtxmprow hlpmsg hlpmsg prepprnt compdef winstr ch2way repasc erabox chccomp repasc erabox chccomp isylim savedata getwname hlpmsg i2chr itypechk ucase leftj left			_
inpfil buildgeo clng isylim cdepend gtxmprow msgbox clng hlpmsg prepprnt cls leftj putstr compdef wnindex rdefg dialog xyz2 chkcomp reset framdef clng chkcomp setmod hlpmsg i2chr itypechk leftj leftg leftgen dialog showme i2chr chkmastr clng drawbo strwin delagen dialog getmod ismpos walldef dialog getmod ismpos windex leftj leftj; leftj leftj leftj leftgen dialog getmod ismpos strwin delagen fucase walldef dialog getmod winstr gtcompgr leftgen sout st syzcoord chkcomp repasc calcdefs chkmastr clng srchf clng clng clng clng srchf stdmsg hlpmsg dialog winchr clng clng gen clng clng clng clng clng clng clng cln			msgbox
isxlim isylim cdepend gtxmprow msgbox clng cls leftj putstr compdef wnindex rdcfg dialog xyz2 repasc erabox chkcomp reset framdef clng savedata getwname hlpmsg i2chr itypechk ucase leftj leftj leftj leftj chrwin putstr chmastr clng showne i2chr clng showne i2chr itypechk leftj left		winstr	chk2way
isylim clng hlpmsg prepprnt cls leftj winindex rdcfg dialog xyz2 repasc erabox chkcomp framdef savedata getwname hlpmsg i2chr i2chr i2chr ucase leftj		buildgeo	clng
msgbox cls leftj putstr compdef wnindex rdofg dialog xyz2 repasc erabox chkcomp reset getwname hlpmsg i2chr seetnod hlpmsg i2chr itypechk setmod hlpmsg i2chr ucase leftj leftj bselect msgbox rightj chrwin putstr chkmastr clng showne delgen drawbo strwin delgen fucase walldef dialog getmod winstr getcompgr ismpos wnindex leftj readke xyzcoord chkcomp repasc calcdefs chimastr sort st clng clng clng srchf cperror cls stdmsg hlpmsg dialog winchr repasc cls bshowne cls getroup bsetmsk cdepend genit repasc cls getroup chrwin erabox hlpmsg chrwin erabox hlpmsg chrwin dialog getroup bsetmsk cdepend genit repasc cls getroup bshowne cls getroup chrwin gendpnd leftj msgbox putstr repasc cls getspred ptustr repasc chrwin gendpnd msgbox petmod putstr repasc stdmsg winchr serabox serabox serabox chrwin gendpnd serabox serabox repasc cls serabox serabox chrwin gendpnd serabox chrwin gendpnd serabox serabox serabox serabox chrwin cmpos gendpnd serabox		cdepend	gtxmprow
prepprnt compdef whindex rdcfg dialog xyz2 repasc erabox chkcomp reset framdef clng savedata getwname hlpmsg i2chr itypechk ucase leftj leftj leftj chkmastr clng showme logen windex stwin delgen dialog getmod winstr gtoompgr leftj compdef cake xyz2 compdef readke xyz2 compdef chkcomp utstr xyz2 compdef sort to chkmastr clng drawbo winstr gtoompgr leftj compdef cake xyzcoord chkcomp repasc calcdefs chkmastr clng sort st clng clng gen bestmak cdepend genit repasc clng gen gen chkmastr clng gen clng gen gen gen gen genit repasc clng gen gen gen genit repasc clng gen gen gen chkmin gendpnd leftj gen getspred pet gemod hlpmsg pet getmod putstr repasc stdmsg getspred ptgroup ptgroup getmod putstr repasc ktdpnd stdmsg ucase msgbox strwin updatgen baspread putstr winstr		clng	hlpmsg
putstr dialog xyz2 repasc erabox chkcomp reset framdef clng savedata getwname hlpmsg setmod hlpmsg i2chr ucase leftj leftj bselect msgbox rightj chrwin putstr chkmastr clng showme clng drawbo strwin delgen fucase walldef dialog getmod winstr gtcompgr ismpos wnindex leftj readke xyzcoord chkcomp repasc calcdefs chkmastr sort st clng clng srchf stdmsg hlpmsg dialog winchr rightj gen bsetmsk cdepend genit repasc clng getspred bshowme clng getspred fucase getspred getspred putstr repasc clng clng srchf stdmsg dialog winchr repasc clng getspred bshowme cls gtcyroup fixed getspred getmod getdpnd msgbox fucase getspred ptgroup putstr readke ktdpnd stdmsg ucase msgbox strwin putstr repasc stdmsg ktwopnd stdmsg ucase msgbox strwin putstr repasc stdmsg ktwopnd stdmsg ucase msgbox strwin putstr repasc stdmsg ktwopnd stdmsg ucase msgbox strwin putstr repasc winchr ptdepend updatgen		cls	_
redefg repasc erabox chkcomp reset framdef clng savedata getwname hlpmsg setmod hlpmsg i2chr showme i2chr itypechk ucase leftj leftj bselect msgbox rightj chrwin putstr chkmastr clng showme clng drawbo strwin delgen fucase walldef dialog getmod winstr gtcompgr ismpos wnindex leftj putstr xyz2 compdef readke Xyzcoord chkcomp repasc calcdefs chkmastr sort st clng clng clng srchf cperror cls stdmsg hlpmsg dialog winchr repasc clng genit repasc clng genit stmow chrwin genit bsotmask cdepend genit chrwin erabox hlpmsg drawbo getdpnd msgbox fucase getmod hlpmsg putstr readke cls gtgroup bshowme cls getspred getmod hlpmsg putstr putstr ktdpnd msgbox stdmsg spread stdmsg ktwdpnd stdmsg ucase msgbox strwin winchr bostness stdmsg ktwdpnd stdmsg ucase msgbox strwin bspread		compdef	
reset getwname hlpmsg i2chr showme i2chr itypechk ucase leftj leftj leftj showme i2chr itypechk ucase leftj leftj leftj showme i2chr itypechk ucase leftj leftj leftj leftj showme i2chr itypechk leftj leftj leftj leftj leftj leftj leftj showme chrwin putstr chkmastr clng showme clng drawbo strwin delgen dialog getmod winstr gtcompgr lismpos wnindex leftj putstr xyz2 compdef readke xyzcoord chkcomp chkcomp repasc calcdefs chkmastr sort_st clng clng clng srchf cperror cls stdmsg hlpmsg dialog winchr repasc clng gen gen bestmsk cdepend genit repasc bshowme cls gtgroup chrwin erabox petpond leftj drawbo getdpnd msgbox getdpnd msgbox getdpnd stdmsg putstr readke ktdpnd stdmsg ucase msgbox strwin updatgen bspread putstr ktdpnd stdmsg ucase winchr putstr winstr	<u> </u>	dialog	
Savedata Setwname	repasc	<u> </u>	
setmod hlpmsg i2chr showme i2chr itypechk ucase leftj leftj bselect msgbox rightj chrwin putstr chkmastr clng showme clng drawbo strwin delgen fucase walldef dialog getmod winstr gtcompgr ismpos wnindex leftj putstr xyz2 compdef readke xyzcoord chkcomp repasc calcdefs chkmastr sort st clng clng clng srchf cperror cls stdmsg hlpmsg dialog winchr rightj gen bsetmsk cdepend genit repasc clng getspred bshowme chrwin erabox hlpmsg cmpos gendpnd leftj drawbo getdpnd msgbox fucase getspred ptgroup getmod putstr repasc stdmsg ktdpnd spread stdmsg getspred putstr repasc stdmsg getspred ptgroup getmod hlpmsg putstr readke ktdpnd spread stdmsg winchr spread stdmsg wtinchr putstr readke ktdpnd spread stdmsg winchr spread stdmsg wtinchr ptdepend updatgen	reset	-	
showme i2chr itypechk ucase leftj leftj bselect msgbox rightj chrwin putstr chkmastr clng showme clng drawbo strwin delgen fucase walldef dialog getmod winstr gtcompgr ismpos wnindex leftj putstr xyz2 compdef readke xyzcoord chkcomp repasc calcdefs chkmastr sort st clng clng srchf cperror cls stdmsg hlpmsg dialog winchr rightj gen bsetmsk cdepend genit repasc clng gespred chrwin erabox hlpmsg cmpos gendpnd leftj drawbo getdpnd msgbox fucase getspred ptgroup getmod hlpmsg putstr readke ktdpnd spread stdmsg winchr crepasc spread stdmsg winch serabox spread strupos getdpnd spread strupos getdpnd spread stdmsg winchr spread stdmsg ktwdpnd spread stdmsg winchr spread stdmsg ktwdpnd spread stdmsg winchr spread stdmsg winchr ptdepend updatgen bspread	savedata		
ucase leftj rightj bselect msgbox rightj chrwin putstr chkmastr clng showme clng drawbo strwin delgen fucase walldef dialog getmod winstr gytcompgr ismpos wnindex leftj putstr xyzcoord chkcomp repasc calcdefs chkmastr sort st clng clng srchf cperror cls stdmsg hlpmsg dialog winchr rightj gen bsetmsk cdepend genit repasc cls gtgroup bshowme cls gtgroup chrwin erabox hlpmsg cmpos getdpnd getspred bshowme getdpnd leftj drawbo getdpnd msgbox fucase getspred ptgroup getmod hlpmsg putstr putstr repasc stdmsg ktwdpnd stdmsg ucase msgbox strwin spread bspread bspread bspread	setmod		
bselect msgbox rightj chrwin putstr chkmastr clng showme clng drawbo strwin delgen fucase walldef dialog getmod winstr gtcompgr ismpos wnindex leftj putstr xyz2 compdef readke xyzcoord chkcomp repasc calcdefs chkmastr sort st cperror clng srchf cperror cls stdmsg hlpmsg dialog winchr rightj gen bsetmsk cdepend genit repasc clng getspred bsetmsk cdepend genit repasc clng getspred bshowme cls gtgroup chrwin gendpnd leftj drawbo getdpnd msgbox fucase getspred ptgroup getmod hlpmsg putstr repasc getspred putstr izchr repasc readke ktdpnd spread stdmsg winchr ptdepend updatgen winchr ptdepend updatgen bspread	showme	— — — ·	
chrwin putstr chkmastr clng showme clng drawbo strwin delgen fucase walldef dialog getmod winstr gtcompgr ismpos wnindex leftj putstr xyz2 compdef readke xyzcoord chkcomp repasc calcdefs chkmastr sort st clng clng srchf cperror cls stdmsg hlpmsg dialog winchr repasc clng gen bsetmsk cdepend genit repasc clng getspred bshowme cls gtgroup chrwin gendpnd leftj drawbo getdpnd msgbox fucase getspred putstr izchr repasc putstr putstr putstr readke ktdpnd spread stdmsg ucase winchr ptdepend updatgen winstr		. 	
clng showme clng drawbo strwin delgen fucase walldef dialog getmod winstr gtcompgr ismpos wnindex leftj putstr xyz2 compdef readke xyzcoord chkcomp repasc calcdefs chkmastr sort st clng clng srchf cperror cls stdmsg hlpmsg dialog winchr repasc clng gen gen posetmsk cdepend genit repasc clng gen genit repasc clng getspred bshowme cls gendpnd leftj drawbo getdpnd msgbox fucase getmod hlpmsg putstr repasc getmod hlpmsg putstr repasc readke ktdpnd spread stdmsg ucase msgbox strwin winstr bspread		-	
drawbo strwin delgen fucase walldef dialog getmod winstr gtcompgr ismpos wnindex leftj putstr xyz2 compdef readke xyzcoord chkcomp repasc calcdefs chkmastr sort st clng clng srchf cperror cls stdmsg hlpmsg dialog winchr rightj gen bsetmsk cdepend genit repasc clng getspred bshowme cls gtgroup chrwin erabox hlpmsg cmpos gendpnd leftj drawbo getdpnd msgbox fucase getspred ptgroup getmod hlpmsg putstr repasc izchr repasc schwin getspred setspred getmod spread stdmsg winchr izchr repasc readke ktdpnd spread stdmsg winchr ptdepend updatgen bspread		-	
fucase walldef dialog getmod winstr gtcompgr ismpos wnindex leftj putstr xyz2 compdef readke xyzcoord chkcomp repasc calcdefs chkmastr sort st clng clng srchf cperror cls stdmsg hlpmsg dialog winchr rightj gen bsetmsk cdepend genit repasc cls gtgroup chrwin erabox hlpmsg cmpos gendpnd leftj drawbo getdpnd msgbox fucase getspred putstr i2chr repasc readke ktdpnd spread stdmsg ucase msgbox strwin winchr ptdepend updatgen bspread		-	_
getmod winstr gtcompgr ismpos wnindex leftj putstr xyz2 compdef readke xyzcoord chkcomp repasc calcdefs chkmastr sort st clng clng srchf cperror cls stdmsg hlpmsg dialog winchr repasc clng genit repasc clng genit repasc clng genit repasc clng getspred bsetmsk cdepend genit repasc cls gtgroup chrwin erabox hlpmsg cmpos gendpnd leftj drawbo getdpnd msgbox fucase getspred ptgroup putstr repasc izchr repasc stdmsg cmpos getmod hlpmsg putstr repasc readke stdmsg ktwdpnd stdmsg ucase msgbox strwin ptdepend updatgen winchr bspread			
ismpos wnindex leftj putstr xyz2 compdef readke xyzcoord chkcomp repasc calcdefs chkmastr sort_st clng clng clng srchf cperror cls stdmsg hlpmsg dialog winchr repasc cls gen bsetmsk cdepend genit repasc cls gtgroup chrwin erabox hlpmsg cmpos gendpnd leftj drawbo getdpnd msgbox fucase getspred getmod hlpmsg putstr putstr readke stdmsg ucase msgbox winchr ptdepend updatgen bspread	_		_
putstr xyz2 compdef readke xyzcoord chkcomp repasc calcdefs chkmastr sort_st clng clng srchf cperror cls stdmsg hlpmsg dialog winchr rightj gen bsetmsk cdepend genit repasc clng getspred bshowme cls gtgroup chrwin erabox hlpmsg cmpos gendpnd leftj drawbo getdpnd msgbox fucase getspred putstr cpasc i2chr repasc putstr readke ktdpnd spread stdmsg ucase msgbox strwin winchr bspread putstr	_		•
readke xyzcoord chkcomp repasc calcdefs chkmastr sort st clng clng srchf cperror cls stdmsg hlpmsg dialog winchr rightj gen bsetmsk cdepend genit repasc clng getspred bshowme cls gtgroup chrwin erabox hlpmsg cmpos gendpnd leftj drawbo getdpnd msgbox fucase getspred ptgroup getmod hlpmsg putstr readke ktdpnd spread stdmsg winchr ptdepend updatgen winchr bspread	-		
repasc calcdefs chkmastr sort_st clng clng srchf cperror cls stdmsg hlpmsg dialog winchr rightj gen bsetmsk cdepend genit repasc clng getspred bshowme cls gtgroup chrwin erabox hlpmsg cmpos gendpnd leftj drawbo getdpnd msgbox fucase getspred ptgroup getmod hlpmsg putstr putstr i2chr repasc stdmsg ktwdpnd spread stdmsg ucase msgbox strwin winchr ptdepend updatgen bspread			chkcomp
sort_st srchf srchf stdmsg winchr bsetmsk cdepend cls cperror cls gen gen genit repasc clng chrwin cmpos drawbo fucase getdpnd fucase getmod putstr readke stdmsg ucase winchr bspread cls clng getspred getspred getdpnd getdpnd msgbox ptgroup ptgroup getmod si2chr readke stdmsg ucase winchr bspread cls getspred getspred getspred ptgroup getmod stdmsg stdmsg stdmsg ucase winchr ptdepend bspread			chkmastr
srchf stdmsg stdmsg winchr bsetmsk repasc clag clag chrwin cmpos drawbo fucase getmod putstr readke stdmsg ucase winchr bspread chrwin getmod putstr stdmsg bspread stdmsg stdmsg stdmsg stdmsd bspread cls dialog gen gen getmod getspred clng getspred ptgroup ptgroup ptgroup ptgroup ptstr repasc stdmsg		clng	
winchr rightj gen bsetmsk cdepend genit repasc clng getspred bshowme cls gtgroup chrwin erabox hlpmsg cmpos gendpnd leftj drawbo getdpnd msgbox fucase getspred ptgroup getmod hlpmsg putstr putstr i2chr repasc readke ktdpnd spread stdmsg ucase msgbox strwin winchr ptdepend updatgen bspread		cperror	
winchr rightj gen bsetmsk cdepend genit repasc clng getspred bshowme cls gtgroup chrwin erabox hlpmsg cmpos gendpnd leftj drawbo getdpnd msgbox fucase getspred ptgroup getmod hlpmsg putstr putstr i2chr repasc readke ktdpnd spread stdmsg ktwdpnd stdmsg ucase msgbox strwin winchr ptdepend updatgen bspread	stdmsg		_
repasc clng getspred bshowme cls gtgroup chrwin erabox hlpmsg cmpos gendpnd leftj drawbo getdpnd msgbox fucase getspred ptgroup getmod hlpmsg putstr putstr i2chr repasc readke ktdpnd spread stdmsg ktwdpnd stdmsg ucase msgbox strwin winchr ptdepend updatgen bspread	-		
bshowme cls gtgroup chrwin erabox hlpmsg cmpos gendpnd leftj drawbo getdpnd msgbox fucase getspred ptgroup getmod hlpmsg putstr putstr i2chr repasc readke ktdpnd spread stdmsg ktwdpnd stdmsg ucase msgbox strwin winchr ptdepend updatgen bspread	bsetmsk		_
chrwin erabox hlpmsg cmpos gendpnd leftj drawbo getdpnd msgbox fucase getspred ptgroup getmod hlpmsg putstr putstr i2chr repasc readke ktdpnd spread stdmsg ktwdpnd stdmsg ucase msgbox strwin winchr ptdepend updatgen bspread	repasc		
cmpos gendpnd leftj drawbo getdpnd msgbox fucase getspred ptgroup getmod hlpmsg putstr putstr i2chr repasc readke ktdpnd spread stdmsg ktwdpnd stdmsg ucase msgbox strwin winchr ptdepend updatgen bspread	bshowme	·	
drawbo getdpnd msgbox fucase getspred ptgroup getmod hlpmsg putstr putstr i2chr repasc readke ktdpnd spread stdmsg ktwdpnd stdmsg ucase msgbox strwin winchr ptdepend updatgen bspread	chrwin		
fucase getspred ptgroup getmod hlpmsg putstr putstr i2chr repasc readke ktdpnd spread stdmsg ktwdpnd stdmsg ucase msgbox strwin winchr ptdepend updatgen bspread			
getmod hlpmsg putstr putstr i2chr repasc readke ktdpnd spread stdmsg ktwdpnd stdmsg ucase msgbox strwin winchr ptdepend updatgen bspread			
putstr i2chr repasc readke ktdpnd spread stdmsg ktwdpnd stdmsg ucase msgbox strwin winchr ptdepend updatgen bspread putstr winstr			•
readke ktdpnd spread stdmsg ktwdpnd stdmsg ucase msgbox strwin winchr ptdepend updatgen bspread putstr winstr	<u>-</u>		-
stdmsg ktwdpnd stdmsg ucase msgbox strwin winchr ptdepend updatgen bspread putstr winstr	<u> </u>		
ucase msgbox strwin winchr ptdepend updatgen bspread putstr winstr		-	
winchr ptdepend updatgen bspread putstr winstr	-		strwin
bspread putstr winstr			updatgen
		rightj	xylimit

Indented Calling Tree for BDAMPREP.EXE in the FACEDAP Preprocess Module (continued)

compprop	putstr	clng
clng	showme	cls
cls	ucase	getspred
erabox	filldp	gtgroup
getspred	clng	ptgroup
gtmat	i2chr	putstr
hlpmsg	rightj	spread
i2chr	fillgeo	geomprnt
leftj	clng	centerln
msgbox	csort2d	clng
ptmat	fndgen	fillgeo
putstr	fndwcomp	leftj
showne	hlpmsg leftj	prephead repasc
spread strwin	rightj	getdata
ucase	fndcomp	getlun
winstr	clng	hlpmsg
conv4prt	cls	getdpnd
clng	erabox	clng
hlpmag	hlpmsg	filldp
cperror	i2chr	fndgen
hlpmsg	leftj	fndwcomp
cprphead	msgbox	hlpmsg
repasc	select	leftj
cprpprnt	fndgen	getinp
clng	clng	addper
cprprep	leftj	chr2i
erabox	fndwcomp	clng
hlpmsg	clng	edtfld
magbox	leftj	fucase
showme	framdef	funits
stdmag	chkcomp	getmod
strwin	clng	putstr
winstr	cls	readke
cprprep	getspred	repasc
centerln	gtgroup	setcur
clng	hlpmsg	setmsk stdmsq
conv4prt	leftj msgbox	strwin
cprphead csort2d	msgbox ptgroup	tone
hlpmsg	putstr	valida
i2chr	readke	winstr
leftj	spread	getspred
repasc	stdmsg	clng
csort2d	wnindex	getlun
delgen	xylimit	hlpmsg
clng	gen	repasc
dellink	chr2i	getvars
leftj	clng	getwalls
dellink	hlpmsg	clng
delwall	i2chr	hlpmag
clng	leftj	iroof
fndwcomp	r2chr	magbox
gtgroup	setcid	select
editform	gendpnd	getwname
optchk	clng	clng erabox
optionb	gtcompgr	hlpmsg
readf	gtxmprow blomea	leftj 🖟
filexist	hlpmsg iroof	magbox
clng erabox	leftj	select
getinp	wnindex	gtcompgr
hlpmsg	xyz2	clng
magbox	genit	leftj
3	•	-

Indented Calling Tree for BDAMPREP.EXE in the FACEDAP Preprocess Module (continued)

	(continued)	
gtfulnam	hlpmsg	updatgen
hlpmsg	leftj	clng
leftj	ptgroup	fndwcomp
gtgroup	chr2i	gen
clng	clng	gtgroup
dellink	hlpmsg	ptgroup
fndwcomp	inslink	wallchk
hlpmsg	leftj	clng
leftj	setcid	hlpmag
rightj	setfid	leftj
gtmat	ptmat	walldef
dellink	clng	clng cls
i2chr	i2chr inslink	delwall
gtxmprow	readf	getlun
clng hlpmsg	addper	hlpmag
leftj	clnq	i2chr
inslink	getinp	leftj
hlpmsg	getmod	magbox
iroof	ismpos	putstr
clng	mkfrm	repasc
hlpmsg	msgbox	samecord
itypechk	putstr	spread
chk2way	readke	ucase
clng	repasc	wallchk
hlpmsg	select	wallok
iroof	setmsk	wallok
wnindex	strwin	clng
ktdpnd	swpcol	hlpmag
clng	winstr	i2chr
ktwdpnd	reset	iroof
clng	repasc	leftj
leftj	rightj	wallrep
leftj	clng	centerln
clng	hlpmsg	clng csort2d
hlpmsg	repasc	hlpmsg
repasc	rsort2d samecord	leftj
optchk	samecolu	prephead
optionb clnq	clud	repasc
erabox	getlun	wnindex
showne	setcid	clng
prephead	clng	gtcompgr
repasc	i2chr	hlpmsg
prepprnt	setfid	leftj
clng	i2chr	xylimit
cprpprnt	spreadop	clng
dpndprnt	calcdefs	hlpmsg
erabox	clng	xy 22
geomprnt	drawbo	hlpmsg
getlun	fndcomp	xyzcoord
getwname	getvars	clng
hlpmsg	getwalls	hlpmsg
leftj	gtcompgr	leftj
magbox	gtfulnam	wnindex
probrep	hlpmag	
showne	leftj	
stdmsg	putstr repasc	
strwin wallrep	repase rightj	
wallrep winstr	showne	
ptdepend	stdmsg	
clng	strwin	
atcompar	winstr	

winstr

gtcompgr

Indented Calling Tree for VALIDFIL.EXE in the FACEDAP Preprocess Module

Midelited Caming 1166 to		
validfil	gtxmprow	wnindex
chgatr	clng	clng
clng	leftj	gtcompgr
cla	iroof	hlpmsg
curof	clng	leftj
curon	hlpmsg	xylimit
getarg	itypechk	clng
getdata	chk2way	hlpmsg
getlun	clng	xyz2
getmod	hlpmsg	hlpmsg
hlpmsg	iroof	xyzcoord
i2chr	wnindex	clng
idtmc	ktdpnd	hlpmsg
imcoff	clng	leftj
isxlim	leftj	wnindex
isylim	clng	
ktdpnd	hlpmsg	
msgbox	repasc	
putstr	onorm	
rdcfg	rightj	
repasc	clng	
review	hlpmsg	
savedata	repasc	
setmod	samecord	
ucase	savedata	
validchg	clng	•
validomp	getlun validchg	
validpnd valid wa l		
validwar	chgloc clng	
	hlpmsg	
bldgchk clng	iroof	
hlpmsg	onorm	
iroof	validomp	
chgloc	clng	
chk2way	i2chr	
clng	leftj	
gtxmprow	validpnd	
hlpmsg	clng	
leftj	gtcompgr	
wnindex	i2chr	
xyz2	ktdpnd	
chkcomp	leftj	
clng	validwal	
hlpmag	bldgchk	
i2chr	clng	
itypechk	leftj	
leftj	wallok	
rightj	validwcm	
compok clng	chkcomp	
fndgen	clng compok	
hlpmsg	leftj	
leftj.	samecord	
fndgen	wnindex	
clng	xylimit	
leftj	xyz2	
getdata	xyzcoord	
getlun	wallok	
hlpmsg	clng	
gtcompgr	hlpmsg	
clng	i2chr	
leftj	iroof	
	leftj	

Indented Calling Tree for MAKEBDMA.EXE in the FACEDAP Analysis Module

```
makebdma
                                            hlpmsg
                                            idtmc
     bdamawri
     bldgchk
                                            imcoff
                                            isxlim
     clng
                                            isylim
     cls
     curof
                                            rdcfq
     curon
                                            setmod
     getarg
                                            ucase
                                      rightj
     getdata
     getlun
                                            clng
     getmod
                                            hlpmag
     hlpmsg
                                            repasc
                                      rsort2d
     idtmc
     imcoff
                                      wnindex
                                            clng
     isxlim
     isylim
                                            gtcompgr
     rdcfg
                                            hlpmsg
                                            leftj
     setmod
                                      xyz2
     ucase
bdamawri
                                            hlpmsg
     clnq
                                      xyzcoord
     getlun
                                            clng
                                            hlpmsg
     gtxmprow
                                            leftj
     hlpmsg
     ktdpnd
                                            wnindex
      leftj
     rsort2d
     wnindex
     xyz2
      xyzcoord
bldgchk
      clng
     hlpmsg
      iroof
getdata
      getlun
      hlpmsg
gtcompgr
      clng
      leftj
gtxmprow
      clng
      leftj
iroof
      clng
      hlpmsg
ktdpnd
      clng
leftj
      clng
      hlpmsg
      repasc
makebdma
      bdamawri
      bldgchk
      clng
      cls
      curof
      curon
      getarg
      getdata
      getlun
```

getmod

Indented Calling Tree for BDAMA.EXE in the FACEDAP Analysis Module

```
bdama
                                    stmrfi
     bdaml
                                    stmswi
bdaml
                                    stowji
     clng
                                    wdbmi
     compd
                                    wdeci
     compr
                                    wdici
     depin
                                    wdrfi
     depmod
                                    wdwli
                              reflect
     getarg
     outpost
                              sideon
     readat
                              stbmi
     sum
                              stcdi
blastpi
                              steci
     reflect
                              stici
     sideon
                              stmrfi
center
                              stmswi
chgloc
                              stowji
compd
                              sum
     blastpi
                              threat
                              wdbmi
     center
     chgloc
                              wdeci
     interp
                              wdici
     onorm
                              wdrfi
compr
                              wdwli
depin
depmod
intcol
interp
intjst
     intcol
intstl
intwd
mapili
marlwi
mar2wi
maulwi
mau2wi
onorm
outpost
rclwi
rc2wi
rcbmi
rceci
rcici
rcmrfi
rcpsi
readat
     mapili
     marlwi
     mar2wi
     maulwi
     mau2wi
     rclwi
     rc2wi
     rcbmi
     rceci
     rcici
     rcmrfi
     rcpsi
     stbmi
     stcdi
     steci
```

stici

Indented Calling Tree for BDAMPOST.EXE in the FACEDAP Postprocess Module

bdampost	putstr	putstr
bload	readke	showne
chgatr	repasc	spread
clng	sort st	strwin
cls	srch f	winstr
curof	stdmsg	editfrm2
curon	winchr	optchk
damagec	bsetmsk	optionb2
dialog	repasc	${\tt readf}$
drawbo	bshowme	fillblst
editfrm2	chrwin	clng
erabox	cmpos	csort2d
genframd	drawbo	hlpmsg
getarg	fucase	leftj
getdata	getmod	rightj
getlun	putstr readke	fillcdam
getmod	stdmsg	clng
hlpmsg i2chr	ucase	csort2d hlpmsg
idtmc	winchr	leftj
imcoff	bspread	rightj
isxlim	chgatr	fillmdam
isylim	chrwin	csort2d
leftj	clnq	hlpmsg
msgbox	curof	leftj
postprnt	curon	rightj
putstr	dialog	framekil
rdcfg	drawbo	clnq
repasc	drawhl	gtxmprow
setmod	drawvl	hlpmsg
showme	erabox	leftj
stdmsg	getinp	wnindex
ucase	getmod	xyz2
bload	hlpmsg	xyzcoord
clng	idtmc	genframd
cls	imcoff	clng
editfrm2	magbox	framekil
erabox	putstr	gtcompgr
fillblst	repasc	gtsortid
getspred	rightj	hlpmag
getwname	spreadop	leftj
hlpmag	strwin winchr	xodgem
leftj	winstr	getdata getlun
msgbox putstr	buildrep	hlpmag
showme	centerin	getspred
spread	clng	clng
strwin	centerln	getlun
winstr	clng	hlpmsg
blstrep	hlpmsg	repasc
centerln	leftj	getwname
clng	repasc	clng
fillblst	csort2d	erabox
leftj	damagec	hlpmsg
repasc	clng	leftj
wrthead	cls	msgbox
bselect	erabox	select
chrwin	filledam	gtcompgr
clng	fillmdam	clng
drawbo	getspred	leftj
fucase	getwname	gtfulnam
getmod	hlpmsg leftj	hlpmsg leftj
ismpos	msgbox	gtsortid
	mayava.	granicia

Indented Calling Tree for BDAMPOST.EXE in the FACEDAP Postprocess Module (continued)

	4
hlpmsg	wdamrep
gtxmprow	centerin
clng	clng
leftj	fillcdam
leftj	leftj
clng	repasc
hlpmsg	wrthead
repasc	wnindex
mdamrep	clng
centerln	gtcompgr
clng	hlpmsg
fillmdam	leftj
leftj	wrthead
wrthead	repasc
optchk	xyz2
optionb2	hlpmag
postprnt	xyzcoord
blstrep	clng
buildrep	hlpmsg
erabox	leftj
getwname	wnindex
hlpmsg	
leftj	
mdamrep	
msgbox	
showme	
stdmsg	
strwin	
wdamrep	
winstr	
rightj	
clng	
hlpmsg	
repasc	
spredop2	
clng	
drawbo	
gtfuln am	
hlpmsg	
putstr	
repasc	
stomsg	
strwin	
winstr	

Indented Calling Tree for COMPREP.EXE in the FACEDAP Preprocess Module

Refer to Appendix C for descriptions of subroutines in Calling Tree followed by "*".

	compprop	b-d2*
bsetmsk	curof	cprphed2* repasc
repasc	curon	cprpprnt
bshowme chrwin	dialog	clng
cmpos	editform	cprprep
drawbo	erabox	erabox
fucase	filxist2*	hlpmag
getmod	getarg	xodpem
putstr	getcomp*	showme
readke	getlun	atdmag
stdmsg	getmod	strwin
ucase	hlpmsg i2chr	winstr
winchr	idtmc	cprprep centerln
bspread	imcoff	
chgatr	initcomp*	clng conv4prt
chrwin	inpfil2*	cprphead
clng	isxlim	csort2d
curof	isylim	hlpmsg
curon	leÎtj	i2chr
dialog drawbo	msgbox	leftj
drawhl	putstr	repasc
drawvl	rdcfg	cselrep
erabox	repasc	centerin
getinp	resetC*	clng
getmod	savecomp*	leftj
hlpmsg	setmod	csort2d
idtmc	showne	dellink
imcoff	ucase	editform
msgbox	compprnt clng	optchk
putstr	cprpprnt	optionb readf
repasc	cselrep	filxist2*
rightj	erabox	clng
spreadop	getlun	erabox
strwin winchr	hlpmsg	getinp
winstr	msgbox	hlpmsg
calcdefs	probrep2*	msgbox
clng	showme	putstr
cperror	stdmsg	showne
hlpmsg	strwin	ucase
rightj	vinstr	fndcomp
centerln	compprop clnq	clng
clng	cls	cls erabox
plbmad	erabox	hlpmsg
leftj	getspred	i2chr
repasc	qtmat	leftj
chgdirl chdir	hlpmsg	msgbox
clng	i2chr	select
dialog	leftj	getcomp*
edtfld	msgbox	getlun
erabox	ptmat	hlpmag
getdir	putstr showme	getdefct*
mkdir	snowne	chr2i
magbox	strwin	hlpmsg getdefmt*
compprep*	ucase	getdermt" chr2i
chgatr	winstr	getinp
chgdirl	conv4prt	addper
clng cls	clng	chr2i
compprat	hlpmsg	clng
Complexed	cperror	edtfld
	hlpmsg	fucase

Revision 1.2

Indented Calling Tree for COMPREP.EXE in the FACEDAP Preprocess Module (Continued)

Refer to Appendix C for descriptions of subroutines in Calling Tree followed by "*".

fun	nits		defint	
get	mod	gts	hort*	
put	str	hlp	msg	
_	dke	put	str	
rep	asc	sho	wine	
	cur	str	win	
set	msk	win	str	
std	msq I	probrep2	*	
	win	cen	terln	
ton	e	cln	g	
val	ida 1	ptmat		
win	str	cln	g	
getsprd2	*	i2c	hr	
cln		ins	link	
get		readf		
ħlp		add	per	
rep	=	cln	g	
getvars2		get	inp	
getwalls			mod	
cln		ism	npo s	
hlp		mk f	:cm	
iro	_ =	msg	zoct	
megi		put	str	
sel		rea	dke	
gtcompgr		rep	asc	
cln			.ect	
left		set	insk	
gtfulnam		str	win	
hlpr		swp	ocol	
left		win	str	
gtmat		resetc		
	lin k	rep	asc	
i2cl	h r	rightj		
gtshort*		cln	ıg	
hlpr	nsq	hlp	msg	
left		rep	asc	
gtxmprow		savecomp	*	
clng	3	chr	:2i	
hlpr	nsg	clr	ng .	
left	٤j¯	get	lun	
initcomp ¹	N	-	omsg	
chri		lef		
inpfil2*		spreador		
clnq	3	cal	Lcd efs	
fcoi		clr		
fdi			awbo	
hlps			dcomp	
sele	ect		vars	
inslink			walls	
hlps	asg		combar	
iroof			fulnam	
clng			pmag	
hlpn	asg		ftj	
leftj			tstr	
clng			pasc	
hlpn			ght j	
repa	isc		owne	
optchk			dmsg	
optionc*			rwin	Revision 1.2
clng	Į.	Wli	nstr	5/20/94
cls				
erab	•			
	comp			
geto	lefct			

Indented Calling Tree for VALIDFL2.EXE in the FACEDAP Preprocess Module

Refer to Appendix C for descriptions of subroutines in Calling Tree followed by "*".

```
getcomp*
      getlun
     hlpmsg
gtshort*
     hlpmsg
     leftj
gtxmprow
     clng
     hlpmsg
     leftj
leftj
     clng
     hlpmsq
     repasc
savecomp*
     chr2i
     clng
     getlun
     hlpmsg
     leftj
validcmp
     clng
     i2chr
     leftj
validfl2*
     chgatr
     chr2i
     clng
     cls
     curof
     curon
     getarg
     getcomp*
     getlun
     getmod
     gtshort*
     gtxmprow
     hlpmsg
     i2chr
     idtmc
    imcoff
    isxlim
    isylim
    leftj
    msgbox
    putstr
    rdcfq
    repasc
    review
    savecomp*
    setmod
    ucase
    validemp
```

Indented Calling Tree for BDAMPSTC.EXE in the FACEDAP Postprocess Module

Refer to Appendix C for descriptions of subroutines in Calling Tree followed by "*".

```
bdampstc*
                                             readke
     bloadc*
                                             stdmsg
     chgatr
                                             ucase
     clng
                                             winchr
     cls
                                       centerln
     comprep*
                                             clng
     curof
                                             hlpmsg
     curon
                                             leftj
     dialog
                                            repasc
     drawbo
                                       compret*
     editfrm2
                                             centerin
     erabox
                                             clng
     getarg
                                       editfrm2
     getcomp*
                                            optchk
     getlun
                                            optionb2
     getmod
                                            readf
     hlpmsg
                                       getcomp*
     i2chr
                                            getlun
     idtmc
                                            hlpmsg
     imcoff
                                       leftj
     isxlim
                                            clng
     isylim
                                            hlpmsg
     leftj
                                            repasc
     msgbox
                                       optchk
     putstr
                                       optionb2
     rdcfg
                                       rightj
     repasc
                                            clng
     setmod
                                            hlpmsg
     showme
                                            repasc
     stdmsg
     ucase
bloadc*
     editfrm2
     erabox
     hlpmsg
     msqbox
     showme
     strwin
     winstr
bselect
     chrwin
     clng
     drawbo
     fucase
     getmod
     ismpos
     putstr
     readke
     repasc
     sort_st
     srchf
     stdmsg
     winchr
bsetmsk
     repasc
bshowme
     chrwin
     cmpos
     drawbo
     fucase
```

getmod putstr Revision 1.2 5/20/94

Appendix B

Description of Primary Subroutines Called in the FACEDAP Code for Building Analysis and Single Component Analysis

NOTE: Subroutines Unique to Single Component Analysis are Shown in Appendix C

```
APPENDIX.B -1
| Saved: 4-28-93 2:37p
1:
         call bdam1
  2:
  3:
          last modified --
  4:
           11/08/91 (tkb)
  5:
           11/19/92 (dds) - added call to outpost; opened file bdama.pst
  6:
  7:
  8:
          purpose --
            sets up building analysis through component damage summation.
  9:
 10:
 11:
          method --
             calls various menu subroutines to obtain desired information
 12:
             about problem to be solved.
 13:
 14:
           input --
 15:
            input data file
 16:
             and tape15 (direct access file with individual element data)
 17:
 18:
 19:
          output --
            tapel5: same file as input
 20:
 21:
 22:
           restrictions
 23:
            none set in bdaml
                            _______
 24:
          call bdamawri ()
 25:
                     writes output from preprocessor in output file in same
 26:
          function:
                      format used for bdama.exe read statements
 27:
                      this file also creates this sortcompg matrix which must
 28:
                      be stored on disk because it links compg in its final
 29:
                      arrangement with the id numbers used by bdama
 30:
                      this info will be needed by post-processor
 31:
 32:
          parameters:
          33:
          call blastpi( r, w, angle, pres, imp)
 34:
 35:
           last modified --
 36:
 37:
            6/11/91 (jpp)
 38:
           purpose --
 39:
             determination of blast pressure and impulse load on a component
 40:
 41:
           method --
 42:
             uses fitted functions from arbrl-tr-02555 (used to fit air blast
 43:
             curves in arlcd-sp-84001) to calculate blast pressure and impulse
 44:
 45:
          input --
 46:
             no files, just variables in argument list
 47:
 48:
           output --
 49:
 50:
            no files, just variables in argument list
 51:
  52:
           restrictions --
            r/w**(1/3) must lie between certain limits which are given in
 53:
             subroutines side-on and reflect
 54:
 55:
 56:
          int = bldgchk (to_screen, lun, fatal_error, total_fatal)
  57:
                      to check wall area connectivity to roof and to place
  58:
          function:
                      the wall index number (from column 14 of wall_table)
 59:
                      of a roof area which is connected to the roofcon
  60:
                      vector. this vector will have same order as wall_table
  61:
                      and it is incorporated into the positions 8 & 9 of
  62:
                      column 15 of wall_table. the roof connectivity is used
  63:
                      in bdamawri.for to get the 'building node'
```

64:

```
APPENDIX.B -2 |
| Saved: 4-28-93 2:37p
                  required by the bdama executable.
  66:
           parameters:
                                        - logical variable indicates where
                         to screen
  67:
                                          error messages will be displayed
  68:
                                          t : display to screen
f : send to file indicated by logical
  69:
  70:
                                              unit lun
  71:
                                        - logical unit number for error output
                         lun
  72:
                            file
  73:
                         fatal error - logical variable indicating if fatal
  74:
                            error occurred. this variable is initialized once
  75:
                            at the beginning of the validation process to false
  76:
                            and set to true each time a fatal error occurs
  77:
                              f - no error occurred
  78:
                              t - fatal error occurred
  79:
                         total fatal - total number of fatal errors
  :08
                         the function name returns the following:
  81:
                         bldgchk: 0 - if all wall areas are connected to at least one roof area
  82:
  83:
                                  1 - if this is not the case
  84:
  85:
            call bload ()
  86:
                        provides a menu for user selection of viewing charge
            function:
  87:
                         weight and its location or viewing the blast load on
  88:
                         building components. data is stored in damage_table
  89:
                         as follows:
  90:
                         non-frame components:
  91:
                         col. 1 - component type in abbreviated form
  92:
                         col. 2 - peak blast pressure to component
  93:
                         col. 3 - peak blast impulse to component
col. 4 - local xl coordinate of component
  94:
  95:
                         col. 5 - local x2 coordinate of component
  96:
                         col. 6 - local y1 coordinate of component
  97:
                         col. 7 - local y2 coordinate of component
  98:
                         col. 8 - distance of component from local (0,0);
  99:
                                  used for sorting the damage table (program
 100:
                                  use only)
 101:
                         frame components:
 102:
                         col. 1 - component type in abbreviated form
 103:
                         col. 2 - peak blast pressure to component
 104:
                         col. 3 - peak blast impulse to component
 105:
                         col. 4 - blastward wall name
 106:
                         col. 5 - local x1 coordinate of component
 107:
                         col. 6 - local x2 coordinate of component
 108:
                         col. 7 - distance of component from local (0,0);
 109:
                                   used for sorting the damage table (program
 110:
                                   use only)
 111:
            parameters:
 112:
                  113:
            call blstrep (wall name)
 114:
                         prints report on blast load for the specified wall.
 115:
                         one report is generated. the report contains the the following information depending on if the wall/roof
 116:
 117:
                          area is a frame:
 118:
 119:
                             non-frame blast load report
 120:
                                component type
 121:
                                blast load peak pressure
 122:
                                blast load impulse
 123:
                                x1 local end or opposite corner point
  124:
                                yl local end or opposite corner point
  125:
                                x2 local end or opposite corner point
  126:
                                y2 local end or opposite corner point
  127:
```

128:

```
APPENDIX.B -3
| Saved: 4-28-93 2:37p
*------
                       frame blast load report
129:
130:
                          component type
                          blast load peak pressure
131:
132:
                          blast load impulse
133:
                          blastward wall name
134:
                          x1 local end or opposite corner point
                         yl local end or opposite corner point
135:
136:
          call select (inbuf, trow, tcol, nrow, numcol, fldwid, -
137:
138:
               numval, option, option no, init_menu, term_key, isort)
139:
          business systems integration
          (512) 680-3940
140:
141:
          copyright (c) 1989, 1989, 1990, 1991, 1992
142:
          all rights reserved
143:
          144:
          call setmsk (mask, type, m len)
145:
          business systems integration
146:
          (512) 680-3940
147:
          copyright (c) 1992
148:
          all rights reserved
          149:
          call showme (funct, row, col, wide, numopt, optl, iopt, -
150:
151:
               init_menu, term_key)
152:
          business systems integration
153:
          (512) 680-3940
          copyright (c) 1988, 1989, 1990, 1991, 1992
154:
155:
          all rights reserved
                          ___________
156:
          call spread (trow, tcol, inbuf, rowhdg, rowlab, colhdg, -
157:
               numrow, numcol, fldwid, tit, dsprow, dspcol, colwid,
158:
                coltyp, icontinue, retopt, edt, maxhead, help_text,
159:
160:
                spread_table, spread_no, max_rows, max_cols, dup)
                        business systems integration (512) 680-3940
161:
162:
                            copyright (c) 1988-1992
163:
164:
                              all rights reserved
165:
166:
                     subroutine
          type:
                     permits input of data in a rectangular format
167:
          function:
                     similar to a spreadsheet
168:
                            top row of data
169:
          parameters: trow
                             top (left) column of data
buffer of items from which the choice is made
170:
                     tcol
171:
                      inbuf
                            header for row labels
172:
                      rowhdg
                     rowlab
                             labels for rows
173:
                     colhdg labels for columns
174:
                     numrow number of rows in the spreadsheet
175:
176:
                     numcol number of columns in spreadsheet
                     fldwid width of each field in the buffer
177:
                              title
178:
                     tit
179:
                      dsprow number of rows displayed
                      dspcol number of columns displayed
180:
                              list of rows in which an edit occurred
181:
                      edt
                      maxhead number of rows used in column header (colhdg)
182:
                     help text line of help text for each column
183:
                     spread no programmer assigned spreadsheet number
184:
185:
                     max rows maximum number of rows in dimension
                     max cols maximum number of columns in dimension
186:
          187:
          call buildgeo ()
188:
189:
          function: allows user definition of the following:
                       1. building wall/roof area
190:
                       2. components geometry definition
191:
192:
                      3. dependencies
```

```
APPENDIX.B -4 |
| Saved: 4-28-93 2:37p
parameters: none
193.
          _____
194:
          call buildrep ()
195:
          function: prints report on building damage
196:
          parameters: none
197:
          .-------------
198:
          call calcdefs (calc_type, row_position, inbuf, max_rows,
199:
                chosen_value, error num)
200:
                      This routine calculates the a value using the eqution
           function:
                      based on calc_type. the value is converted to character
201:
 202:
                       form and returned in chosen value. this routine is
 203:
                      called by spreadop to calculate the default value in
 204:
                       a field on a spreadsheet. error checking is performed
 205:
                       to test for a blank field or a field containing a 0.
 206:
           parameters:
 207:
                                   - contains the number of the equation
                       calc type
 208:
                                    to be used in the calculation
 209:
                       row_position - current row on spreadsheet
 210:
                                   - character array containing spreadsheet
 211:
                                    values
 212:
                                   - maximum row in array used to store
                       max rows
 213:
                                     spreadsheet data, not necessarily the
 214:
                                     number of rows displayed on the
 215:
                                     spreadsheet. there could be less rows
 216:
                                     displayed
 217:
                       chosen_value - returns values selected by user or
 218:
                                    calculated by program
 219:
                                   - error flag indicator
                       error num
 220:
                                     0 : no error occurred
 221 .
                                     1 : error, field was blank
 222:
                                     2 : error, field was zero
 223:
           224:
           call cdepend (wall_name, wall_index)
 225:
                       allows user computation and editing of dependenices for
           function:
 226:
                       the specified wall/roof area. user selects/enters from
 227:
                       a list the following for each component in depend_table:
 228:
                       column 1 : component type - i.e. concrete slab, concrete
 229:
                                  beam, etc. (display only)
 230:
                       column 2 : local x1 coordinates of end opposite corner
 231:
                                 points of the component (display only)
 232:
                       column 3 : local y1 coordinates of end opposite corner
 233:
                                 points of the component (display only)
 234:
                       column 4 : local x2 coordinates of end opposite corner
 235:
                                 points of the component (display only)
 236:
                       column 5 : local y2 coordinates of end opposite corner
 237:
                                 points of the component (display only)
 238:
                       column 6 : dependent component id number (display only)
 239:
                        column 7-10 :
 240:
                                  supporting component number(s) - up to 4
 241:
                                  component numbers which will be pre-set
 242:
                                  but may be edited by the user.
 243:
           parameters:
 244:
                                   - name of wall/roof area user selected
                       wall name
 245:
                          for component definition
 246:
                        wall index - index into the wall area used to
 247:
                        access wall number for current wall name
 248:
            249:
            call center(x1,x2,x3,x4,xc)
  250:
  251:
            last modified --
  252:
             6/1/90 \text{ (map)}
  253:
  254:
            purpose --
```

determines the center of an element

255:

256:

```
| Saved: 4-28-93 2:37p
  _____
257:
258:
          method --
259:
          see comments in code
260:
261:
          input --
262:
          arguments:
             x1,x2,x3,x4 coordinates for element end points
263:
264:
265:
          output --
266:
           arguments:
267:
                          coordinates for element center point
            жÇ
268:
269:
           restrictions
270:
           2-way elements must be approximately rectangular
271:
          272:
          call centerln (in line, out_line, line_length)
273:
274:
                     centers the line, in line with a line that line length
          function:
                     long and places it centered in out line. the routine
275:
                     may be called with the same variable for in line and
276:
277:
                     out line.
          ______
278:
279:
          call charge (ier)
280:
                    allows user input and editing of the load definition.
          function:
                     the load definition consists of the charge weight in
281:
                      lbs and the charge global coordinates in feet. the
282:
                      form uses load buf to store character data.
283:
                     one form is used and its input values are as follows:
284:
285:
                       loaddef - form 2
286:
                                 charge weight (ft)
                                 charge location in global x direction
287:
                                 charge location in global y direction
288:
                                 charge location in global z direction
289:
290:
          parameters:
291:
                     ier
                                 - 0 : no errors occurred
                                   1 : error occurred converting data from
292:
293:
                                       character to real. field was not
294:
                                       entered
          ______
295:
296:
          call chgdirl (ier, directory name, directory path)
                   business systems integration copyright (c) 1990
297:
298:
                   all rights reserved
299:
300:
                     change or make directory
          function:
301:
          parameters:
302:
303:
          call chgloc(xchg,xc,vn,r,angle)
304:
305:
           last modified --
306:
           6/4/90 \text{ (map)}
307:
308:
           purpose --
           calculate distance to charge and angle of blast wave impact
309:
310:
311:
           method --
312:
           see comments in code
313:
          subroutine child (prog, commd, icls)
                               314:
          call chk2way (to screen, row, lun, fatal_error,
315:
                error occurred, total fatal, total warnings)
316:
                     perform validity of 2-way reinforced masonry element that were previously done in bdama subroutine mar2wi.
317:
          function:
318:
                      the following items are checked:
319:
                       1. endpoints are diagonal.
320:
```

```
APPENDIX.B -6
| Saved: 4-28-93 2:37p
                 2. aspect ratio is within reason.
321 .
           parameters:
322:
                                      : logical variable indicating where
                        to screen
723
                                        error messages go
324:
                                          t - messages will go to screen
325:
                                          f - message go to file
326:
                                      : row number in compg of 2-w reinforced
327:
                        IOW
                                        masonry element to be checked
328:
                                      : logical unit number of file messages
329:
                        lun
                                        are written to if to_screen if false
330:
                                      : logical variable indicating if fatal
331:
                        fatal error
                                        error occurred. this variable is
332:
                                        initialized once at the beginning of
333:
                                        the validation process to false
334:
                                        and set to true each time a fatal
335:
                                        error occurs
336:
                                          f - no error occurred
337:
                                          t - fatal error occurred
338:
                                      : total number of fatal errors
                        total fatal
339:
                        total_warnings : total number of warnings
340:
341:
           342:
           call chkcomp (wall name, row, mat_name, comp_type_name, -
343:
                 comp_prop_name, component_id, to_screen,
344:
                  lun, fatal_error, total_fatal,
345:
                  total_warnings, ier)
346:
                        checks if the specified component type, comp_type_name,
347:
           function:
                        is valid for the given material name, mat_name. the
348:
                        component property name is then check to make sure it
349:
                        is valid for the specified component type. this
350:
                        routine is used by the preprocessor, bdamprep, and
351:
                        the validation program, validfil. if used by the
352:
                        preprocessor error message go to the screen. if used
353:
                        by the validation program, error messages go to the
354:
                        file specified by lun. the following checks are made:
355:
                          1. component type defined
356:
                          2. component property name defined
357:
                          3. material-component type mismatch
358:
                          4. 1-way slabs and panels, beams and joists and
359:
                             exterior columns are checked to insure the component
360:
                             is linear. this is a fatal condition in validfil.
361:
                          5. metal stud walls and masonry 1-way components are
362:
                             checked to see if the component is contained on
363:
                             a wall area. a warning is issued if it is not.
 364:
                          6. metal steel joists are checked for the component
 365:
                             being on a roof. a warning is issued if it is not.
 366:

    exterior columns and wood walls are checked to

367:
                             ascertain if they are on a wall. a fatal error
368:
                             in validfil occcurs if this is not the case.
 369:
                          8. wood roofs are check to make sure the component
 370:
                             is on a roof area. this is a fatal condition for
 371:
                             validfil.
 372:
                          9. two-way components are checked for diagonal
 373:
                             endpoints and reasonable aspect ratio. failure
 374:
                             of one of these 2 criteria results in a fatal
 375:
                             condition.
 376:
                         10. two-way masonry components are tested for being
 377:
                             on a roof. if this component is on a roof a
 378:
                             warning is issued.
 379:
                         11. interior columns are checked make sure end 2
 380:
                             coordinates are blank. the coordinates are
 381:
                             blanked by the program if they contain values.
 382:
                         12. interior columns are checked to insure that end 1
 383:
                             is contained on a roof area. a fatal error results
 384:
```

```
_____
                                                      APPENDIX.B -7 |
| Saved: .4-28-93 2:37p
if end 1 is on a wall.
385;
                        13. frame components are tested to makes sure the
386:
                            blastward area is a wall. a roof area results
387:
388:
                            in a fatal error.
                        14. component type-component property mismatch
389:
                        15. component in component geometry found in
390:
                            component properties
391:
392:
          parameters:
                                   : name of current wall
393:
                       wall name
                       row -
                                     : current row in compg that is being
394:
                                 processed
: name of material type
395:
396:
                       mat name
                       comp_type_name : name of component type
397:
398:
                       comp_prop_name : component property name
399:
                       component id : 9 character id number for component
                                      defined by the specified mat name,
400:
                                       comp type name, and comp_prop_name
401:
                                     : logical variable indicating where
402:
                       to screen
                                       error messages go
403:
                                         t - messages will go to screen
404:
                                         f - message go to file
405:
                                     : logical unit number of file messages
                       lun
406:
407:
                                       are written to if to screen if false
                                     : logical variable indicating if fatal
408:
                       fatal_error
                                       error occurred. this variable is
409:
                                       initialized once at the beginning of
410:
                                       the validation process to false
411:
                                       and set to true each time a fatal
412:
                                       error occurs
413:
                                         f - no error occurred
414:
                                         t - fatal error occurred
415:
                       total_fatal : total number of fatal errors
416:
                       total_warnings : total number of warnings
417:
                                     : error flag
418:
                       ier
                                         0 - no error occurred
419:
                                         1 - material not found
420:
                                         2 - material-component type mismatch
421:
                                         3 - component property name not
422:
 423:
                                            defined
                                         4 - component type-component
 424:
                                            property mismatch
 425:
                                         5 - no component properties defined
 426:
                                            for component type
 427:
                                         6 - component type not defined
 428:
                                         7 - error occurred in itype check
 429:
 430:
           431:
           call chkmastr (wall_name)
 432:
                       checks If any previous master component have been
 433:
           function:
                       changed to unique. if this type of change occurred
 434:
                       the user is warned and given the chance to make
 435:
                       the component a master again and thus not lose the
 436:
                       generated components associated with it.
 437:
 438:
           parameters:
                      wall name : name of current wall
 439:
 440:
           call compd(ndepa,idepa,xb,xchg,w,nr,dam,iwrit)
 441:
 442:
            last modified --
 443:
 444:
             11/08/91 (tkb)
             11/24/92 (dds) - increase dimesion of dam columns from 6 to 8 and
 445:
             put pbar and ibar in those columns 12/01/92 (dds) - commented out write to screen
 446:
 447:
             01/28/93 (dds) - removed cp2, ci2 from web steel joist read of direct
 448:
```

```
APPENDIX.B -8
| Saved: 4-28-93 2:37p
access file; removed idc from frames; added debuggin
449:
                              prints
450:
451:
           purpose --
452:
             determines percent damage to building components
 453:
 454:
 455:
            uses p-i curves in section 5.0 of blast vulnerability guide
 456:
 457:
            input --
 458:
             arguments:
 459:
                             number of dependency pairs in array idepa
 460:
               ndepa
                             array with pairs of independent/dependent elements
 461:
               idepa
                             two dimensional array with building orientation
 462:
               жb
                             coordinates
 463:
                             coordinates with charge location
 464:
               xchq
 465:
            output --
 466:
 467:
             arguments:
                             array with component damage levels and
 468:
               dam
                             repair/replace factors
 469:
 470:
            restrictions
 471:
             none set in compd
 472:
 473:
 474:
           call compdef (wall name, wall index)
 475:
                        allows user input of the components which will define
 476:
           function:
                        the specified wall/roof area. user selects from a list
 477:
                        the following for each component:
 478:
                          1. component material type - concrete, steel,
 479:
                                 masonry or wood (selection field)
 480:

    component type - i.e. concrete slab, concrete

 481:
                                 beam, etc. (selection field)
 482:
                              component property name - user defined component
                          3.
 483:
                                 which was defined in the component properties
 484:
                                 definition phase. (selection field)
 485:
                              component id name - based on wall index,
 486:
                          4.
                                 number of component property name and number
 487:
                                 generated. the generated is 00 is master and
 488:
                                 a sequential number if generated.
 489:
                                  (display only field)
 490:
                          5. x1 end or opposite corner points of the component
 491:
                                  (user entry)
 492:
                              yl end or opposite corner points of the component
                           6.
 493:
                                  (user entry)
 494:
                              yl end or opposite corner points of the component
 495:
                                  (user entry)
 496:
                              yl end or opposite corner points of the component
                           8.
 497:
                                  (user entry)
 498:
                              master component selection
                           9.
 499:
                                 yes - component is a master and repeated groups
 500:
                                       will be generated from it
 501:
                                    - component is unique - no components will
 502:
                                       be generated from it
 503:
                                  (selection field)
 504:
                              center to center spacing (user entry)
                          10.
 505:
                          11. local direction
 506:
                                      - component generation will be in the
 507:
                                       positive x direction
 508:
                                      - component generation will be in the
 509:
                                       positive x direction
 510:
                                  (selection field)
 511:
                          12. number of additional repeated groups generated
 512:
```

	4-28-93 2:37p APPENDIX.B -9
3:	from the master (user entry)
4:	13. though not displayed on spreadsheet, this column
5:	temporarily contains the generate flag so it
6:	it not lost thru deletion from table when
	component retrieved. (program use)
7:	component retrieved. (program use)
8:	14-17 used by program for maintaining independent
9:	component id with the corresponding dependent id.
0:	 check if any master components where changed to
1:	unique components while on spreadsheet. user
2:	is prompted to restored component to master
3:	status or the leave as a unique component and
4 :	have any generated components deleted.
5:	2. component type defined
	3. component property name defined
<u>6</u> :	
7:	4. material-component type mismatch
8:	5. 1-way slabs and panels, beams and joists and
9:	exterior columns are checked to insure the component
0:	is linear. this is a fatal condition in validfil.
1:	metal stud walls and masonry 1-way components are
2:	checked to see if the component is contained on
3:	a wall area. a warning is issued if it is not.
	7. metal steel joists are checked for the component
4:	being on a roof. a warning is issued if it is not.
5:	8. exterior columns and wood walls are checked to
<u>6</u> :	
7:	ascertain if they are on a wall. a fatal error
8:	in validfil occcurs if this is not the case.
9:	9. wood roofs are check to make sure the component
0:	is on a roof area. this is a fatal condition for
1:	validfil.
2:	10. two-way components are checked for diagonal
2: 3:	endpoints and reasonable aspect ratio. failure
	of one of these 2 criteria results in a fatal
4:	
5:	condition.
6:	11. two-way masonry components are tested for being
7:	on a roof. if this component is on a roof a
8:	warning is issued.
9:	12. interior columns are checked make sure end 2
0:	coordinates are blank. the coordinates are
1:	blanked by the program if they contain values.
2:	13. interior columns are checked to insure that end 1
2: 3:	is contained on a roof area. a fatal error results
4:	if end 1 is on a wall.
5:	14. frame components are tested to makes sure the
6:	blastward area is a wall. a roof area results
7:	in a fatal error.
8:	15. component type-component property mismatch
9:	16. component in component geometry found in
0:	component properties
0: 1:	17. component coordinates are checked to make sure
	that coordinates are not outside wall coordinates
2:	*****
3:	parameters:
4:	wall_name - name of wall/roof area user selected
5:	for component definition
6:	wall index - index into the wall area used to
7:	access wall number for current wall_name
8:	
9:	call compok (wall name, row, comp_type_name, component_id,-
	lun, fatal_error, total_fatal, -
0:	Iun, lacal error, cocal lacar,
1:	total warnings, error occurred)
2:	function: checks if the specified component has all numeric
3:	values defined and if it is a master that its slaves
4:	have been generated and at least one slave exists.
4 .	
5:	parameters:

.

```
APPENDIX.B -10 |
| Saved: 4-28-93 2:37p
                      : current row in compg that is being
577:
                                       processed
578:
                       comp_type_name : name of component type
579:
                                    : 9 character id number for component
                       component_id
580:
                                       defined by the specified mat_name,
581:
                                       comp_type_name, and comp_prop name
582:
                                      : logical unit number of file messages
                       lun
583:
                                       are written to if
584:
                                      : logical variable indicating if fatal
                       fatal error
585:
                                       error occurred. this variable is
586:
                                       initialized once at the beginning of
587:
                                       the validation process to false
588:
                                       and set to true each time a fatal
589:
                                       error occurs
590:
                                         f - no error occurred
591:
                                         t - fatal error occurred
592:
                       total fatal
                                     : total number of fatal errors
593:
                       total warnings : total number of warnings
594:
                       error_occurred : logical variable indicating if a
595:
                                       error occurred in the checking
596:
                                       done by this subroutine
597:
598:
           call compprop ()
599:
           function: allows user selection of component type and then
 600:
                       allows the definition of the required properties for
 601:
                       that type. the four types are concrete, steel,
 602:
                       masonry and wood. the spreadsheet number corresponds to
 603:
                       the icnindex of the component type.
 604:
 605:
           call compr(dam)
 606:
 607:
            last modified --
 608:
             7/15/91 (jpp)
 609:
             2/5/93 (cjo) add prestressed beams
 610:
 611:
 612:
            purpose --
             determines reusablilty of building components
 613:
 614:
 615:
            in blast vulnerability guide, section 5.0, 5.1,6.0
 616:
 617:
 618:
            input --
 619:
 620:
 621:
            output --
 622:
             ?
 623:
 624:
            restrictions --
 625:
            none set in compr
 626:
           call conv4prt (comp_table, irow, itype, imat, icomp, rval)
 627:
                        convert all real values in the specified row in
 628:
                        comp_table from character to real and places them
 629:
                        in rval.
 630:
           parameters:
 631:
                        comp table : character array containing the component
 632:
                                     properties for a specified component type.
 633:
                                   : row in comp_table containing values to
                        irow
 634:
                                     be converted
 635:
                                   : type of component property being printed
                        itype
 636:
                                     (1-24)
 637:
                                   : material type
                        imat
 638:
                                   : component type
                        icomp
 639:
                                   : output real array containing converted
                        rval
 640:
```

```
APPENDIX.B -11
| Saved: 4-28-93 2:37p
641:
                                     real values
642:
           function: prints error messages for the cell default
                        calculations in spreadop. this routine could
643:
644:
                        be expanded to provide more specific error messages
                        but for now only two generic message will be printed
645:
           parameters: none
646:
647:
           call cprphead (lun, itype, report, line kt)
648:
                       writes column headers for the preprocessor component
649:
650:
                        property reports based on itype and report and
                        increments the line counter, line kt, based on number
651:
652:
                        of lines written.
653:
                           itype
                                            property
                                                           # reports
654:
                                                               2
                             1
                                            rcbmi
655:
                             2
                                            rclwi
                                                               2
                             3
                                            rc2wi
656:
                                                               2
657:
                             4
                                            rceci
                             5
                                                               2
658:
                                            rcici
                                                              2
                             6
                                            rcmrfi
659:
                             7
                                                               2
660:
                                            rcpsi
661:
                             8
                                            stbmi
                            9
                                                               1
662:
                                            stmswi
                            10
                                            stowji
                                                               1
663:
                                            stcdi
                            11
664:
665:
                           12
                                            steci
                           13
                                            stici
666:
                                                               2
667:
                           14
                                            stmrfi
                                                              2
                           15
                                             maulwi
668:
                                                              1
                           16
                                            mau2wi
669:
                                                               2
670:
                            17
                                             mar1wi
671:
                            18
                                            mar2wi
                                                               2
                            19
672:
                                            mapili
                                                               2
                            20
                                             wdwli
673:
674:
                            21
                                             wdrfi
                                                               2
675:
                            22
                                             wdbmi
676:
                            23
                                             wdeci
                                             wdici
677:
                            24
678:
           parameters:
                                : logical unit connected to report file
679:
                        lun
                                : type of header to be printed
680:
                        itype
681:
                        report
                               : indicates which report to print for the
                                  specified itype (1 or 2)
682:
                        line kt : current number of lines written to report file
683:
684:
                 call cprpprnt (all)
685:
686:
                       allows user to select and print reports based on
           function:
687:
                        selection of material type and component type or print
                        all component reports.
688:
689:
           parameters: all : logical variable indicating if all reports are
690:
                               to be printed
                               t - print all reports
f - show menu for user selection
691:
692:
693:
694:
           call cprprep (itype, imat, icomp, report, tot_rep, comp_name)
           function: writes component property report for specified itype
695:
696:
                        and report number.
697:
                           itype
                                            property
                                                           # reports
                                            rcbmi
698:
                            1
                                                               2
                             2
                                            rclwi
699:
                                                               2
                             3
700:
                                            rc2wi
                             4
                                                              2
                                            rceci
701:
                                                              2
702:
                             5
                                            rcici
703:
                                             rcmrfi
704:
                                             rcpsi
```

```
APPENDIX.B -12
| Saved: 4-28-93 2:37p
                                           stbmi
                            8
705:
                                            stmswi
                            9
706:
                                            stowji
                            10
707:
                                                              2
                                            stcdi
                            11
708:
                                                              2
                                            steci
                            12
709:
                                                              2
                                            stici
                            13
710:
                                            stmrfi
                           14
711:
                                                              2
                                            maulwi
                           15
712:
                                                              1
                                            mau2wi
                           16
 713:
                                                              2
                                            marlwi
                            17
 714:
                                                              2
                                            mar2wi
                            18
 715:
                                            mapili
                            19
 716:
                                                              2
                                            wdwli
                            20
 717:
                                                              2
                                             wdrfi
                            21
 718:
                                                              2
                                             wdbmi
                            22
 719:
                                                              2
                                             wdeci
                            23
 720:
                                             wdici
                            24
 721:
           parameters:
 722:
                                 : type of header to be printed
                        itype
 723:
                                 : material type
                        imat
 724:
                                 : component type
                        icomp
 725:
                                  : number of report to be printed (1 or 2)
                        report
 726:
                                 : total reports that will be printed for
                        tot rep
 727:
                                    the current component type
 728:
                        comp_name : component property name of current report
 729:
            call csort2d (array, nval, isortcol, numrow, numcol, idir, - .
 730:
 731:
                  iconvert, ier)
 732:
            business systems integration
 733:
            (512) 680-3940
 734:
            copyright (c) 1988, 1989, 1990, 1991, 1992
 735:
            all rights reserved
 736:
 737:
                        sort a character 2 dimensional array
            function:
 738:
            parameters: array character array to sort
 739:
                                 number of values to sort
                        nval
 740:
                        isortcol column in array sort is based on
 741:
                                 number of rows to sort
                        numrow
 742:
                                 number of columns in array
                         numcol
 743:
                                  sort direction (1 ascending, -1 descending)
                         idir
                         iconvert flag indicating whether to convert to real
 744:
 745:
                                   for the column specified in doing the sort
 746:
                                   0: sort the column as character
 747:
                                   1: convert the number to real and sort as
 748:
                                     if column is real
 749:
                                   0 : no errors occurred during sort
                         ier
 750:
                                   1 : isortcol exceeded number of columns in
  751:
                                      array
  752:
  753:
            call damagec (damaged)
  754:
            function: provides a menu for user selection of viewing the most
  755:
                         damaged components or all components.
  756:
            parameters:
  757:
                                       : logical variable which indicates if
                         damaged
  758:
                                        building sustained any damage
  759:
                                         t : building sustained damage
  760:
                                         f : building sustained no significant
  761:
                                            damage (i.e. no component's damage
  762:
                                            exceeded 0%)
  763:
  764:
             call delgen (mindex, id, previous)
  765:
            function: deletes all generated components associated with
  766:
                         the specified master id
  767:
            parameters:
  768:
```

```
*----*
| Saved: 4-28-93 2:37p
                                                APPENDIX.B -13
mindex : index to where generated components begin
                               for the specified master
770:
                             : id number of master component which will
                     id
771:
                               used to find generated components to delete
772:
                     previous : pointer to previous row processed
773:
774:
         call dellink (link, free, head, tail, current, previous)
775:
                    deletes the specified component indicated by current
776:
         function:
                     from the linked list and adds it back to the free list.
777:
         parameters:
778:
                             - link list used for maintaining order in
                     link
779:
                                a given storage area
780:
                              - pointer to first free row in storage area
781:
                     free
782:
                              - pointer to first used row in storage area
                     head
                              - pointer to last used row in storage area
                     tail
783:
                     current - pointer to current row being processed
784:
                     previous - pointer to previous row processed
785:
         ______
786:
          call delwall (wall name)
787:
                    deletes all master and unique components assoicated
          function:
788:
                     with the specified wall, wall_name, from the compg
789:
                     storage area.
790:
791:
         parameters:
                    wall name - name of current wall/roof area
792:
793:
794:
          call depin(ndep,idep,nr)
795:
          last modified --
796:
797:
           6/11/91 (jpp)
798:
799:
          purpose --
           input, check validity of dependency list.
800:
801:
802:
          method --
           2-dimensional array idep contains a list of
803:
                   independent element id, dependent element id.
804:
           the element id's are coded to contain material id and component id
805:
806:
           where
                   element id = (10000*imat) + (1000*icomp) + id
807:
808:
809:
          input --
          arguments:
810:
                        total number of dependencies
811:
           ndep
                        array with pairs of independent and dependent components
812:
            idep
                        three dimensional array with record numbers in
813:
            nr
                        tape15 where component data is stored. see
814:
                        description in bdam.
815:
816:
817:
          output --
           input arguments (ndep, idep) and:
818:
819:
          restrictions
820:
          none set in depin
821:
           822:
823:
          call depmod(ndep,idep,ndepa,idepa)
824:
           last modified --
825:
826:
           11/08/91 (tkb)
827:
          purpose --
828:
           modify dependency array to include multiple stepped dependencies.
829:
           for example, if the following pairs are entered: 1,2
830:
 831:
                     2,3
832:
```

```
APPENDIX.B -14 |
| Saved: 4-28-93 2:37p
then the pair 1,3 should be added.
833:
834:
           method --
835:
            see comments in code
836:
837:
          input --
838:
839:
            arguments:
                         total number of dependencies entered
840:
             ndep
                         array with pairs of independent and dependent components
841:
             idep
                         entered by user
842:
843:
           output --
844:
            arcuments:
845:
                         total number of dependencies after modification
             ndepa
846:
                         array with all pairs of independent and dependent component
             idepa
847:
848:
849:
           restrictions
           none set in depmod
850:
851:
          call dpndprnt (wall name)
852:
                      prints report on dependencies for the specified
853:
          function:
                      wall. one report is generated. the report contains the
854:
                      the following information.
855:
856:
                         non-frame dependency report 1
857:
                            component type
858:
                            x1 local end or opposite corner point
 859:
                            yl local end or opposite corner point
860:
                            x2 local end or opposite corner point
 861:
                            y2 local end or opposite corner point
862:
                            dependent component id #
 863:
                            independent component id # (1-4)
 864:
           865:
           call editform (form_file_name, form_number, usrdat,
 866:
                key_pressed)
 867:
                     : editform
           routine
 868:
                   : displays the specified form, form_file_name, to the
 869:
           function
                       screen and allows user processing of that form
 870:
           parameters : form file name - name of form, with full path, to be
 871:
                         edited_
 872:
 873:
                                    - number of form to be edited
                       form number
 874:
                                    - array containing form data
                       usrdat
 875:
                                   - the key stroke the user left the
                       key_pressed
 876:
                                     form with
 877:
           878:
           call editfrm2 (form_file_name, form_number, usrdat,
 879:
                key_pressed)
 880:
           routine
                     : edtfrm2
 881:
                     : displays the specified form, form_file_name, to the
 882:
           function
                       screen and allows user processing of that form for
 883:
                       the bdam postprocessor
 884:
           parameters : form_file_name - name of form, with full path, to be
 885:
                          edited
 886:
 887:
                                    - number of form to be edited
                       form number
 888:
                                     - array containing form data
                       usrdat
 889:
                                    - the key stroke the user left the
                       key_pressed
 890:
                                      form with
 891:
 892:
           call filexist (input_file_name, ier)
 893:
           routine : filexist
 894:
           function : allows user to select the bdamprep file name to save
 895:
                       the file under. the default is the current file name.
 896:
```

```
| Saved: 4-28-93 2:37p
                                                       APPENDIX.B -15 |
*------
                       issues warning message if the selected file already
897:
                       exists. user then has the choice of writing over the
898:
899:
                       existing file or aborting the save.
900:
          parameters :
901:
                       input file name
                                          : current preprocessor file name
902:
                                           selected by user. contains full
903:
                                           path
904:
                       ier
                                          : 0 - save file
                                          : 1 - do not save file
905:
906:
907:
           call fillblst (wall name, number damaged)
                     fills the damage_table with required data for displaying
908:
           function:
909:
                       all components for a specified wall area.
910:
                       data is stored in damage_table as follows:
911:
                       non-frame components:
912:
                       col. 1 - component type in abbreviated form
913:
                       col. 2 - peak blast pressure to component
914:
                       col. 3 - peak blast impulse to component
                       col. 4 - local x1 coordinate of component
915:
916:
                       col. 5 - local x2 coordinate of component
917:
                       col. 6 - local yl coordinate of component
918:
                       col. 7 - local y2 coordinate of component
                       col. 8 - distance of component from local (0,0);
919:
920:
                                used for sorting the damage table (program
921:
                                use only)
922:
                       frame components:
                       col. 1 - component type in abbreviated form
923:
                       col. 2 - peak blast pressure to component
924:
                       col. 3 - peak blast impulse to component
925:
                       col. 4 - blastward wall name
926:
927:
                       col. 5 - local x1 coordinate of component
                       col. 6 - local x2 coordinate of component
928:
                       col. 7 - distance of component from local (0,0);
929:
930:
                                used for sorting the damage table (future
                                program use if decided to perform a secondary
931:
932:
                                sort)
933:
          parameters:
934:
                       wall name
                                    : current wall/roof area name
935:
                       number damaged : total number of most damaged components
936:
                                for specified wall
937:
938:
           call fillcdam (wall name, number damaged)
939:
                       fills the damage_table with required data for displaying
           function:
940:
                       components for a specified wall area.
941:
                       data is stored in damage table as follows:
942:
                       non-frame components:
943:
                       col. 1 - component type in abbreviated form
944:
                       col. 2 - local x1 coordinate of component
                       col. 3 - local x2 coordinate of component
945:
946:
                       col. 4 - local y1 coordinate of component
                       col. 5 - local y2 coordinate of component
947:
948:
                       col. 6 - percent damage to component
                       col. 7 - repair or replace flag to component
949:
                       col. 8 - p-i diagram terms pbar
950:
951:
                       col. 9 - p-i diagram terms ibar
952:
                       col.10 - distance of component from local (0,0);
953:
                               used for sorting the damage table (program
954:
                                use only)
955:
                       frame components:
                       col. 1 - component type in abbreviated form
956:
                       col. 2 - percent damage to component
957:
                       col. 3 - repair or replace flag to component
958:
                       col. 4 - p-i diagram terms pbar
959:
960:
                       col. 5 - p-i diagram terms ibar
```

```
APPENDIX.B -16
| Saved: 4-28-93 2:37p
col. 6 - blastward wall name
961:
                        col. 7 - local x1 coordinate of component
962:
                        col. 8 - local x2 coordinate of component
963:
                        col. 9 - distance of component from local (0,0);
964:
                                 used for sorting the damage table (future
965:
                                 program use if decided to perform a secondary
966:
                                 sort)
967:
           parameters:
968:
                                       : user assigned name of wall/roof area,
969:
                        wall name
                                         unless the area is a frame in which
970:
                                         case the program assigned the area name
971:
                                         of frame
972:
                        number_damaged : total number of most damaged components
973:
                                for specified wall
974:
975:
           call filldp (mindex, dindex, blankit)
976:
                       copies the data required for the dependency spreadsheet
977:
           function:
                        depend table from compg. note: if storage of compg
978:
                        or depend table changes, indexes on the do loops will
979:
                        have to change.
980:
981:
           parameters:
                                     - row index of compg table to be filled
                        mindex
982:
                                     - row index of depend table to be filled
                        dindex
983:
                                     - logical variabe indicating if old
                        blankit
984:
                                       dependencies are to be copied or blanked
985:
                                       t : blank old dependencies in compg and
986:
                                           depend table
987:
                                       f : copy old dependencies from compg to
 988:
                                           depend table
 989:
 990:
            991:
           call fillgeo (wall_name, num_print, ier)
 992:
                       retrieve a master/unique data and generated components
            function:
 993:
                        for the specified wall name from compg and places in
 994:
                        the work table for printing. for non-frame items
 995:
                        the distance from the local 0,0 to the endpoints is
 996:
                        placed in column 14 of work_table for each row. for
 997:
                        frames the is placed in column 8. these distances
 998:
                         will be used for sorting.
 999:
            parameters:
1000:
                                     - user assigned wall name which is
1001:
                         wall name
                                       to be displayed with dependencies
1002:
                                     - number of components put in work table
                         num print
1003:
                                       to be printed
1004:
                                     - 0 : no errors occurred
                         ier
1005:
                                       1 : no componets found for wall
1006:
                                       2 : number of components exceeded
1007:
                                           numrow
1008:
1009:
            call fillmdam (number_most_damaged, frames)
1010:
                      fills the damage table with required data for displaying
            function:
1011:
                        most damaged components.
1012:
                         non-frame components:
1013:
                         col. 1 - wall name component is on
1014:
                         col. 2 - component type in abbreviated form
1015:
                         col. 3 - percent damage to component
1016:
                         col. 4 - local xl coordinate of component
1017:
                         col. 5 - local x2 coordinate of component
1018:
                         col. 6 - local y1 coordinate of component col. 7 - local y2 coordinate of component col. 8 - p-i diagram terms pbar
1019:
1020:
1021:
                         col. 9 - p-i diagram terms ibar
1022:
                         frame components:
1023:
                         col. 1 - blastward wall name (has * in 1st column
1024:
```

```
| Saved: 4-28-93 2:37p
                                                          APPENDIX.B -17
                                 to distinguish from non-frame components
1025:
                         col. 2 - component type in abbreviated form
1026:
                         col. 3 - percent damage to component
1027:
                         col. 4 - local x1 coordinate of component
1028:
                         col. 5 - local x2 coordinate of component
1029:
                         col. 6 - p-i diagram terms pbar
1030:
                         col. 7 - p-i diagram terms ibar
1031:
1032:
           parameters:
                         number most components : total number of most damages
1033:
                                                  components
1034:
                                                : logical variable
1035:
                         frames
                                                  t - frame components were among
1036:
1037:
                                                     most damaged
                                                  f - no components were among
1038:
                                                     most damaged
1039:
                               ______
1040:
1041:
            call fndcomp (mat name, comp type name, comp prop name, ier)
                       provides pop-up menu of list of defined component
            function:
1042:
                         property names based on the current material type
1043:
                         and component type. the selected name is returned in
1044:
1045:
                         comp_prop_name along with an error flag.
1046:
           parameters:
1047:
                        mat name
                                       : name of material type
                         comp type name : name of component type
1048:
                         comp_prop_name : selected component property name
1049:
                                       : 0 - no error occurred
1050:
                                          1 - material name or component name
1051:
                                              is not defined
1052:
                                          2 - material name not found in material
1053:
                                              list - programmer error
1054:
                                          3 - component name not found in
1055:
1056:
                                              component list - programmer error
                                          4 - no component properties defined
1057:
                                              for specified material and component
1058:
                                              type
1059:
                                          5 - user aborted selection of component
1060:
                                             property name
1061:
1062:
1063:
            call fndgen (m id, mindex, found)
1064:
1065:
            function:
                        searches the generated link list for where data for
                         specified master component begins.
1066:
1067:
1068:
            parameters:
                         part id
1069:
                         mindex
1070:
                         found
1071:
1072:
            call fndwcomp (wall_name, mlink, head, mindex, previous,
1073:
                   itype, found, id)
1074:
                         searches the specified link list for where data for
1075:
            function:
                         specified wall name begins. if type = 1, the master/
1076:
                         unique list is searched until the specified wall_name
1077:
                         if found. the pointer, mindex then returns the row
1078:
                         number of the found record; found is set to true and
1079:
                         previous returns the row number in the list before
1080:
                         the found record. if type = 2, the generated
1081:
                         list is also searched for the specified wall name. once
1082:
                         the wall is found the search switches to searching
1083:
                         for the id of the master the components where generated
1084:
                         from. found is set to true, if data for
1085:
                         specified wall is found. found is false if not found.
1086:
                         mindex returns the row number of the first wall name
1087:
                         component in the list. previous returns the row
1088:
```

```
APPENDIX.B -18
| Saved: 4-28-93 2:37p
number in the list before the record was found.
1089:
1090:
         parameters:
1091:
                                  : name of current wall
                      wall name
1092:
                                   : pointer to head of linked list
                      head
1093:
                                  : pointer to where data for the specified
                      mindex
1094:
                                     wall begins
1095:
                                   : pointer to previous row processed
                      previous
1096:
                                   : indicates if storing master/unique,
1097:
                      itype
                                     generated or frame components
1098:
                                     1 - master/unique component
1099:
                                     2 - generated component
3 - frame
1100:
1101:
                                    : logical variable that indicates
1102:
                      found
                                     finding of desired data
1103:
                                     t - components found for specified
1104:
                                         wall
1105:
                                      f - no components found for specified
1106:
                                         wall
1107:
                                    : if itype = generated, id contains
                      id
1108:
                                     the number of the master id the
1109:
                                     components were generated from
1110:
           1111:
          call framdef ()
1112:
                    allows user input of the frame components. user selects
          function:
1113:
                      the following items indicate the columns used in the
1114:
                      work area, frame table:
1115:
                        1. frame type - reinforced concrete frame or
1116:
                              steel frame
1117:
                        2. component property name - user defined component
1118:
                              which was defined in the component properties
1119:
                              definition phase. (selection field)
1120:
                        3. frame id - same format as component id but to
1121:
                               indicate this is a frame the first portion is
1122:
                              00. the second number is based on a sequential
1123:
                              number assigned for the frames and last portion
1124:
                              is 00. (display only field)
1125:
                        4. wall containing frame which is nearest the blast.
1126:
                               (selection field)
1127:
                        5. x1 coordinate of ground level endpoint of column
1128:
                               (user entry)
1129:
                        6. yl coordinate of ground level endpoint of column
1130:
                               (user entry)
1131:

    total height of frame (user entry)

1132:
                        8. material type - concrete or steel, dependent on
1133:
                           component type selected (program use)
1134:
1135:
           parameters:
           1136:
           call framekil (rownum, depcomp, numdcomp)
1137:
                       finds all components within the area of influence
1138:
                       of a frame and sends the array depoomp with the id numbers
1139:
                       of all these components back to the calling
1140:
                       program
1141:
           parmeters:
1142:
                                  - the row number of frame component
                       rownum
1143:
                                   compg matrix
1144:
                                  - an array with id numbers for all
                       depcomp
1145:
                                   components which are failed due
1146:
                                   to failure of frame in rownum
1147:
                                 - number of component id numbers in
                       numdcomp
1148:
                                   depcomp
1149:
           1150:
           call gen (wall_index, set_gen_flag, w_comp_tmp, ier)
 1151:
           function: generates the requested components from the master
 1152:
```

Saved:	4-28-93 2:37p APPENDIX.B -19
153:	component.
154:	parameters:
155:	<pre>wall index - index into the wall area used to</pre>
156:	access wall number for current wall name
157:	set_gen_flagtrue. : indicates that generated
-	set_gen_fragtrue. : indicates that generated
158:	components were created from the specified master
.159:	and that the gen_flag parameter in associated with
160:	the compg storage area should be updated to reflect
161:	that components were generated from that master.
162:	w comp tmp - temporary storage for the row containing
163:	the master component that components are to be
164:	generated from. this is done since the work space,
1165:	work_table, is actually the same for master/unique
166:	components and for the generated components.
167:	ier - 0 : no error occurred
168:	1 : one or more required pieces of data
169:	in missing from the current master
170:	
171:	call gendpnd (wall name, wall index, total depend)
172:	function: finds independent component id numbers for each compone
	this is a 'first-cut' method in that it is based on a
.173:	
174:	number of assumptions discussed in user's manual
.175:	parmeters:
.176:	wall_name - name of wall/roof area user selected
177:	for component definition
178:	wall index - index into the wall area used to
.179:	access wall number for current wall_name .
180:	total_depend - total number of components in current
181:	wall
182:	
183:	call conframd ()
	call genframd ()
184:	function: generated dependencies for frame. finds all frames
185:	which have a damage of 100% and then find all component
.186:	within the volume of the frame and sets them to
187:	total failure (100%) also. recalculate building damage
188:	building repair/replace factor and % reusable floor.
189:	parameters:
190:	
191:	call genit (wall_name, wall_index, id, num_gen, w_comp_tmp,-
192:	term key, ier)
193:	function: allows user to view/ edit generated components from
	the master component user collect Controlled Link
194:	the master component . user selects/enters from a
195:	list the following for each component:
196:	 component material type ~ concrete, steel,
197:	masonry or wood (selection field)
198:	2. component type - i.e. concrete slab, concrete
199:	beam, etc. (selection field)
L200:	3. component property name - user defined component
201:	which was defined in the component properties
202:	definition phase. (selection field)
203:	4. component id name - based on wall index,
204:	number of component property name and number
L205:	generated. the generated is 00 is master and
L206:	a sequential number if generated.
207:	(display only field)
208:	5. xl end or opposite corner points of the component
1209:	(user entry)
L210:	f. yl end or opposite corner points of the component
1211:	(user entry)
	7. yl end or opposite corner points of the component
212.	
1212:	
.213:	(user entry)

```
_____
                                                     APPENDIX.B -20 |
| Saved: 4-28-93 2:37p
wall name - name of wall/roof area user selected
                        for component definition
1218:
                       wall_index - index into the wall area used to
1219:
                        access wall number for current wall name
1220:

    id number of component which generated

1221:
                                    components
1222:
                       num_gen - number of additional repeat groups generated
1223:
                                 for current master component
1224:
                       w_comp_tmp- temporary storage for the row containing
1225:
                                  the master component that components are
1226:
                                  to be generated from. this is done since
1227:
                                  the work space, work table, is actually
1228:
                                  the same for master/unique components and
1229:
                                  for the generated components.
1230:
                       term_key - key_stroke user exited on
1231:
                       ier
                                - 0 : no error occurred
1232:
                                 1 : error occurred opening sprdhead.dat
1233:
                                    or reading file
1234:
1235:
           call geomprnt (wall_name)
1236:
           function: prints report on component geometry for the specified
1237:
                       wall. the report contains the following information
1238:
                       depending on if the wall/roof area is a frame:
1239:
1240:
                          non-frame component geometry report 1
1241:
                             item #
1242:
                             material type
1243:
                             component type
1244:
                             component property name component id #
1245:
1246:
1247:
                          non-frame component geometry report 2
1248:
                             xl local end or opposite corner point
1249:
                             yl local end or opposite corner point
1250:
                             x2 local end or opposite corner point
1251:
                             y2 local end or opposite corner point
1252:
                             master component generation (yes or no)
1253:
                             center to center spacing
1254:
                             local direction of c/c spacing (x or y)
1255:
                             number of components to be generated
1256:
1257:
                          frame component geometry report
1258:
                             component type
1259:
                             component property name
1260:
                             component id #
1261:
                             blastward wall name
1262:
                             x1 local end coordinate
1263:
                             yl local end coordinate
1264:
                             height of frame
1265:
1266:
           call getdata (input_file_name)
1267:
           function: retrieve data for current problem which is required
1268:
                       for a restart of the preprocessor.
1269:
           parameters:
1270:
                       input_file_name : name of file to retrive data
1271:
                                          from; contains full path
1272:
           1273:
           call getdpnd (wall_name, numrow, blankit, total_dpnd, ier)
1274:
                      retrieve a master/unique data and generated components
           function:
1275:
                       for the specified wall name from compg and places in
1276:
                       the depend table for spreadsheet display.
1277:
           parameters:
1278:
                                   - user assigned wall name which is
                       wall name
1279:
                                     to be displayed with dependencies
```

1280:

```
_____
| Saved: 4-28-93 2:37p
                                                          APPENDIX.B -21
- maximum number of rows which can be
1281:
                         numrow
                                       displayed on spreadsheet
1282:
                                      - logical variabe indicating if old
1283:
                         blankit
                                        dependencies are to be copied or blanked
1284:
                                        t : blank old dependencies in compg and
1285:
                                            depend table
1286:
1287:
                                        f : copy old dependencies from compg to
1288:
                                            depend_table
1289:
                         total dpnd
                                      - total number of components for current
                                        wall/roof area
1290:
                                      - 0 : no errors occurred
1291:
                         ier
                                        1 : no componets found for wall
1292:
1293:
                                        2 : number of components exceeded
1294:
                                           numrow
1295:
            call getinp (row, pcol, prompt, lprmpt, col, mtype, mlen, -
1296:
                  outstr, term, clegal, numter)
1297:
1298:
            business systems integration
1299:
            (512) 680-3940
            copyright (c) 1988, 1989, 1990, 1991, 1992
1300:
1301:
            all rights reserved
1302:
            call getspred (spread_no, ier)
1303:
                        reads all required variables for the specified
1304:
                         spreadsheets from the data file sprdhead.dat.
1305:
                         the following values are mead:
1306:
1307:
                         1. spread no
                         2. number of columns on spreadsheet
1308:
                         3. number of rows on spreadsheet
1309:
                         4. number of column header lines
1310:
                         5. column headers
1311:
1312:
                         6. help text
                         7. variable type for each column (i.e. character,
1313:
1314:
                               integer, real)
                         8. number of defaults
1315:
1316:
                         9. column numbers of columns which have default settings
                        10. default value for the default columns
1317:
                        11. fields where duplication on spreadsheet allowed
1318:
                            0 - field can be duplicated
1319:
                            1 - field can not be duplicated
1320:
1321:
            parameters:
                                    - number of spreadsheet to retrieve data for
1322:
                        spread no
                                    - 0 : no error occurred
1323:
1324:
                                      1 : error occurred opening file,
                                         sprdhead.dat
1325:
                                      2 : error occurred reading sprdhead.dat
1326:
1327:
            call getvars (spread_no, col_position, calc_type)
function: used by spreadop for determining the columns used in
1328:
1329:
                         default calculations. based on the spreadsheet number,
1330:
                         spread no and the column position on the spreadsheet,
1331:
                         col position, the required column numbers are returned
1332:
1333:
                         along with an indicator of the equation to be used for
                         calcluating.
1334:
1335:
            call getwalls (wall name, term key, ier)
function: provides pop-up menu of list of defined wall areas
1336:
1337:
                         (roofs are excluded) for user selection. the selected
1338:
1339:
                         name is returned along with the keystroke the user
                         exited the menu with.
1340:
1341:
            parameters:
                         wall name
                                      : name of chosen wall/roof area
1342:
                                      : reflects user's decision to continue
                         term_key
1343:
1344:
                                        processing or return to previous menu
```

```
APPENDIX.B -22 |
| Saved: 4-28-93 2:37p
                              *-----
                                     0 - continue processing
                                      f2 (316) - return to previous menu
1346:
                                     : 0 - no error occurred
                        ier
1347:
                                      1 - no wall areas defined
1348:
                                      2 - or user aborted selection
1349:
1350:
1351:
           call getwname (wall_name, wall_index, itype, term key, ier)
1352:
                        provides pop-up menu of list of defined wall/roof
1353:
           function:
                        areas. the selected name is returned along with
1354:
                        the keystroke the user exited the menu with.
1355:
           parameters:
1356:
                                    : name of chosen wall/roof area
                        wall name
1357:
                        wall index : index into wall data for wall/roof
1358:
                                      area selected
1359:
                                     : indicator of whether to display message
                        itype
1360:
                                      applying to wall component definitions
1361:
                                       or dependencies
1362:
                                       1 - component definitions
1363:
                                       2 - dependencies
1364:
                                       3 - blast load on building components
1365:
                                       4 - most damaged components
1366:
                                       5 - component geometry print
1367:
                                     6 - dependency print: reflects user's decision to continue
1368:
                        term key
1369:
                                       processing or return to previous menu
1370:
                                       0 - continue processing
1371:
                                       f2 (316) - return to previous menu
1372:
                                     : 0 - no error occurred
                        ier
1373:
                                       1 - no wall areas defined or user
1374:
                                           aborted selection
1375:
1376:
1377:
            call gtcompgr (comp_id, irow, ier)
1378:
                        uses the specified component id to determine the
1379:
                        row number of the component property in compg storage
1380:
1381:
                        area.
1382:
            parameters:
                                      : component id number used for searching
                        comp_id
1383:
                                        the compg array
1384:
                                      : row number of component in compg area
1385:
                        irow
                                      : 0 - no error occurred
                        ier
1386:
                                         1 - not found in compg table
1387:
1388:
1389:
            call qtfulnam (comp type name, icomp, imat, ier)
1390:
                        uses the specified component property name to find
1391:
            function:
                        the icomp and imat that corresponds to the specified
1392:
1393:
                        type.
            parameters:
1394:
                        comp_type_name : name of component type
1395:
                         icomp : component # with a given material group
1396:
                                       : material #
                         imat
1397:
                                       : 0 - no error occurred
1398:
                         ier
                                        1 - component type name not found
1399:
1400:
            call gtgroup (wall_name, itype, id)
1401:
            function: retrieves the wall/roof roof group specified by wall name
1402:
                         of master and unique components, if itype = 1, from
1403:
                        the compg storage array and places them in w_comp_table
1404:
                         for spreadsheet display. the generate flag is
1405:
                        maintained aligned with the correct component by
1406:
                        removing it from compg and place it in an unused col,
1407:
                         column 13, of the w_comp_table. if itype = 2, generated
1408:
```

```
*-----
| Saved: 4-28-93 2:37p
                                                         APPENDIX.B -23
  1409:
                         components are being retrieved from compg and places
                         in gencomp. if itype = 3, frame componets are being
1410:
1411:
                         retrieved from compg and placed in frame table.
1412:
                         compg contains all required information for each
1413:
                         component used in the building. the rows within compg
1414:
                         are never physically moved but order is maintained
1415:
                         thru the link list mlink. pointers to the beginning
                         and end of the used master/unique components are
1416:
1417:
                         maintained in head and tail. pointers to the
                         beginning and end of the used generated nodes are
1418:
1419:
                         maintained in ghead and gtail. pointers to the
1420:
                         beginning and end of the frame nodes are maintained in
1421:
                         fhead and gtail.
1422:
1423:
                         compg contains the following for itypes 1 & 2:
1424:
                         non-frame components:
1425:
                         column 1 : wall name
1426:
                         column 2 : component material type (wood, concrete, etc)
1427:
                         column 3 : component type (r/c beam, etc)
1428:
                         column 4 : component property name (user assigned)
1429:
                         column 5 : component id number
1430:
                         column 6 : x1 coordinates of end or corner
1431:
                         column 7 : yl coordinates of end or corner
1432:
                        column 8 : x2 coordinates of end or corner
column 9 : y2 coordinates of end or corner
1433:
                         column 10: master component indicator
1434:
1435:
                                   yes - master
1436:
                                   no - unique
1437:
                         column 11: center to center spacing
1438:
                         column 12: local direction
1439:
                                   local x - generate in positive x direction
                        local y - generate in positive y direction column 13: number of additional repeated components
1440:
1441:
1442:
                        column 14: independent component 1
1443:
                        column 15: independent component 2
1444:
                        column 16: independent component 3
1445:
                        column 17: independent component 4
1446:
1447:
                        compg contains the following for itype 3:
1448:
                        frame components
1449:
                        column 1 : wall name - always frame
1450:
                        column 2 : component material type (concrete or steel)
1451:
                        column 3 : component type (r/c frame or steel frame)
                        column 4 : component property name (user assigned)
1452:
1453:
                        column 5 : component id number
                        column 6 : x1 coordinate of blastward exterior column,
1454:
1455:
                                   coordinates are in the local coordinates of
1456:
                                   the blastward wall
1457:
                        column 7 : yl coordinate of blastward exterior column,
1458:
                                   coordinates are in the local coordinates of
1459:
                                   the blastward wall
1460:
                        column 8 : blank
1461:
                        column 9 : blank
1462:
                        column 10: set to no since no components may be generated
                                   from this component
1463:
1464:
                        column 11: name of blastward wall
1465:
                        column 12: total frame height
1466:
1467:
           parameters:
1468:
                        wall_name - name of current wall/roof area
1469:
                        itype - indicates if storing master/unique,
1470:
                                     generated or frame components
1471:
                                     1 : master/unique component
1472:
                                     2 : generated component
```

```
APPENDIX.B -24
| Saved: 4-28-93 2:37p
                                 3 : frame components
                                - id number of master component which will
1474:
                                  used to find components generated from
1475:
                                  that master (used when itype = 2)
1476:
           1477:
           call gtmat (imat, icomp)
1478:
                      retrieves the component material property specified
           function:
1479:
                       by imat and icomp from the xmp storage array and places
1480:
                       them in comp_table for spreadsheet display. xmp
1481:
                       contains all required information for component
1482:
                       material property used in the building. the rows
1483:
                       within xmp are never physically moved but order is
1484:
                       maintained thru the link list clink.
1485:
1486:
                       xmp contains the following:
1487:
                       column 1 : icn - identification based on material
1488:
                                       and selected subset component
1489:
                       column 2 : component property name (user assigned)
1490:
                       column 3 : weighting factor
1491:
                       column 4 : beam width
1492:
                       column 5 : beam thickness
1493:
                       column 6 : loaded width
1494:
                       column 7 : total weight
1495:
                       column 8 : compressive concrete strength
1496:
                       column 9 : steel yield strength
1497:
                       column 10: depth to tensile steel
1498:
                       column 11: area tensile steel
1499:
                       column 12: moment of inertia
1500:
                       column 13: boundary condition
1501:
           parameters:
1502:
                                 - material indicator
1503:
                        imat
                                  1 : concrete
1504:
                                   2 : steel
1505:
                                   3 : masonry
1506:
                                  4 : wood
1507:
                                 - number of subcategory which occurs under
                        icomp
1508:
                                  each material
1509:
1510:
           call gtsortid (compg_row, sort_row, ier)
function: uses the specified row number in compg to search
1511:
1512:
                        the sortcompg table to determine the index into the
1513:
                        damage table, dam, from bdama.
1514:
           parameters:
1515:
                                      : row number of current component in
                        comp row
1516:
                                       the compg array
1517:
                                      : row number of component in sortcompg
                        sort row
1518:
                                       and dam
1519:
                                      : 0 - no error occurred
                        ier
1520:
                                        1 - not found in sortcompg table
1521:
1522:
            ______
1523:
            call gtxmprow (mat_name, comp_type_name, comp_prop_name,
1524:
                  irow, ier)
1525:
                        uses the specified material type, mat_name, the
1526:
            function:
                        specified component type, comp_type_name, and the
1527:
                        specified component property name to determine the
1528:
                        row number of the component property in xmp storage
1529:
                        area.
1530:
1531:
            parameters:
                                   : name of material type
                        mat name
1532:
                        comp type name : name of component type
1533:
                        comp_prop_name : name of component property name
1534:
                        irow : row number of component in xmp area ier : 0 - no error occurred
1535:
1536:
```

```
*-----
| Saved: 4-28-93 2:37p
                                                    APPENDIX.B -25
1537:
                                      1 - invalid material type
                                      2 - specified component type not
1538:
1539:
                                         valid for specified material
1540:
                                      3 - specified component property not
1541:
                                         valid for specified component type
1542:
1543:
          call inslink (link, free, head, tail, current, ier)
1544:
1545:
          function:
                      removes the first available free space from the free
                      list and inserts it to the tail of the linked list.
1546:
1547:
          parameters:
1548:
                      link
                               - link list used for maintaining order in
1549:
                                 a given storage area
1550:
                      free
                                - pointer to first free row in storage area
1551:
                      head
                                - pointer to first used row in storage area
                      tail
1552:
                                - pointer to last used row in storage area
                               - pointer to current row being processed
1553:
                      current
                               - 0 : no error occurred
1554:
                      ier
                                 1 : no more free space available
1555:
                                     in storage area - it's full
1556:
1557:
          1558:
          call interp(imat,icomp,pbar,ibar,dmg,n)
1559:
1560:
            last modified --
1561:
              10/28/91 (tkb)
              2/8/93 (cjo) asymptotes for all components added into interp
1562:
                     new asymptotes added for components steel components
1563:
1564:
                     without tension membrane and rc components with arching
                     wood comp. asymptotes changed to correlate to new p-i
1565:
1566:
                     parameters
1567:
1568:
            purpose --
1569:
              determination of percent damage to component to be used in level
1570:
              of damage evaluation
1571:
            method --
1572:
             numerically defines relevant p-i curves for material/component
1573:
1574:
              type under consideration, and compares these values with calculated
1575:
              pbar and ibar terms to identify the corresponding level of damage.
1576:
            input --
1577:
1578:
             imat
1579:
              icomp
1580:
              pbar
1581:
              ibar
1582:
1583:
           output --
1584:
             dmg
1585:
1586:
           restrictions --
             none set in interp
1587:
1588:
                               1589:
          int = iroof (wall index)
1590:
                      to determine if wall area in row 'wall index'
           function:
                      is a 'roof'. roof returns a 1 if the area is a roof
1591:
1592:
                      otherwise it returns a 0.
1593:
          parameters:
                      wall index - index into the wall area used to
1594:
                        access wall number for current wall name
1595:
1596:
          call itypechk (to_screen, row, wall_name, comp_type,
1597:
                 comp_id, imat, icomp, lun, fatal_error,
error_occurred, total_fatal, total_warnings)
1598:
1599:
          function: checks for component specific errors based on the itype
1600:
```

*	4-28-93 2:3		APPENDIX.B -26 /
* 1601:			nt. the following conditions are checked:
1602:			bs and panels, beams and joists and
1603:			columns are checked to insure the component
1604:		ie linear	. a warning is issued if it is not.
1605:			d walls and masonry 1-way components are
1606:			o see if the component is contained on
1607:		a wall ar	ea. a warning is issued if it is not.
1608:			el joists are checked for the component
1609:			a roof. a warning is issued if it is not.
1610:		A exterior	columns and wood walls are checked to
1611:		ascertain	if they are on a wall. a fatal error
1612:			il occours if this is not the case.
1613:			s are check to make sure the component
1614:			oof area. this is a fatal condition for
1615:		validfil.	
1616:			omponents are checked for diagonal
1617:			and a minimum aspect ratio. failure
1618:		of one of	these 2 criteria results in a fatal
1619:		condition	
1620:			asonry components are tested for being
1621:			. if this component is on a roof a
1622:			s issued.
1623:			columns are checked make sure end 2
1624:			es are blank. the coordinates are
1625:			y the program if they contain values.
1626:			columns are checked to insure that end 1
1627:			ned on a roof area. a fatal error results
1628:			is on a wall.
1629:			ponents are tested to makes sure the
1630:			area is a wall. a roof area results
1631:		in a fata	
1632:	parameters:		- +
1633:	*	to screen	: logical variable indicating where
1634:			error messages go
1635:			t - messages will go to screen
1636:			f - message go to file
1637:		IOM	: current row in compg that is being
1638:			processed
1639:		wall name	: name of wall/roof area user selected
1640:		_	for component definition
1641:		comp type	: name of component type
1642:		comp_id	: 9 character id number for component
1643:			defined by the specified mat_name,
1644:			comp_type_name, and comp_prop_name
1645:		imat	: number between 1 and 4 which indicates
1646:			the material type of the current
1647:			component
1648:		icomp	: number between 1 and 7 which indicates
1649:			the current component number in the
1650:			given material group
1651:		lun	: logical unit number for error output
1652:			file
1653:		fatal_error	: logical variable indicating if fatal
1654:			error occurred. this variable is
1655:			initialized once at the beginning of
1656:			the validation process to false and
1657:			set to true each time a fatal error
1658:			occurs
1659:			f - no error occurred
1660:			t - fatal error occurred
1661:		error_occurred	: indicates error occurred in this
1662:			category
1663:		total_fatal	: total number of fatal errors which have
1664:			occurred

```
_____
| Saved: 4-28-93 2:37p
                                                        APPENDIX.B -27 |
                     total warnings : total number of warnings
1665:
1666:
           call ktdpnd (total depend)
1667:
1668:
          function: checks the compg area to count the total number of
1669:
                        dependencies pairs that exist.
1670:
           call ktwdpnd (wall_name, total_depend)
1671:
1672:
           function:
                     checks the compg area to count the total number of
1673:
                        independent components that exist for the current wall
1674:
           call leftj (chosen value, char variable, variable_length)
1675:
1676:
           function:
                        left justifies data within a variable. char_variable
                        contains the data to be right justified. chosen value
1677:
1678:
                        contains the left justified data. variable should
1679:
                       not be longer than 80 characters
1680:
1681:
           call mapili (ncl,nel)
1682:
1683:
            masonry pilasters
1684:
1685:
           last modified --
             7/01/91 (jpp)
1686:
            01/28/93 (dds) - made moment of inertia be input, xim, rather than
1687:
1688:
                             calculated value, im; added debugging prints
1689:
           purpose --
1690:
1691:
             data is read from and/or written to a direct access file.
1692:
1693:
            in blast vulnerability guide, see section 5.0, figure 5.1*
1694:
            and section 6.1
1695:
1696:
1697:
            input --
            tapel5: direct access file with individual element data.
1698:
1699:
1700:
            output --
            tapel5 and arguments (see input discussion)
1701:
1702:
1703:
            restrictions
1704:
            none set in mapili
1705:
1706:
           call marlwi (ncl,nel)
1707:
1708:
            reinforced masonry one way walls
1709:
            last modified --
1710:
1711:
             7/01/91 (jpp)
1712:
            01/28/93 (dds) - made moment of inertia be input, xim, rather than
                             calculated value, im; removed a from read and write;
1713:
                             added debugging prints, weight/length in ci is
1714:
1715:
                             calculated using areal weight and section width
1716:
            purpose --
1717:
            data is read from and/or written to a direct access file.
1718:
1719:
1720:
            method --
1721:
             in blast vulnerability guide, see section 5.0, figure 5.1*
1722:
             and section 6.1
1723:
1724:
            input --
            tapel5: direct access file with individual element data.
1725:
1726:
1727:
            output --
             tapel5 and arguments (see input discussion)
1728:
```

```
APPENDIX.B -28 |
| Saved: 4-28-93 2:37p
1729:
           restrictions
1730:
           none set in marlwi
1731:
           _____
1732:
           call mar2wi(ncl,nel)
1733:
1734:
           reinforced masonry two way walls
1735:
1736:
            last modified --
1737:
             7/02/91 (jpp)
1738:
            01/28/93 (dds) - made moment of inertia be input, xim, rather than
1739:
                            calculated value, im; added debugging prints
1740:
                            modified gamma in ci into effective wall density
1741:
1742:
            purpose --
1743:
             data is read from and/or written to a direct access file.
1744:
1745:
1746:
            in blast vulnerability guide, see section 5.0, figure 5.1*
1747:
             and section 6.1
1748:
1749:
            input --
1750:
             tapel5: direct access file with individual element data.
1751:
1752:
            output --
1753:
             tape15 and arguments (see input discussion)
1754:
1755:
1756:
            restrictions
            none set in mar2wi
1757:
                      ______
1758:
           call maulwi (ncl,nel)
1759:
1760:
            unreinforced masonry one way walls
1761:
1762:
            last modified --
1763:
1764:
             7/03/91 (jpp)
             01/28/93 (dds) - made moment of inertia be input, xim, rather than
1765:
                             calculated value im; made z input rather than calculated
1766:
                             value; added iarch; and added debugging prints; removed
1767:
                             cross-sectional area, a, from input list
1768:
                             modified ci to get weight/length in terms of gamma
1769:
1770:
             purpose --
1771:
             data is read from and/or written to a direct access file.
1772:
1773:
             method --
1774:
             see theory manual
1775:
1776:
1777:
             input --
             tapel5: direct access file with individual element data.
1778:
 1779:
 1780:
             tapel5 and arguments (see input discussion)
 1781:
 1782:
 1783:
             restrictions
             none set in maulwi
 1784:
                                         1785:
            call mau2wi(nc1, nel)
 1786:
 1787:
             unreinforced masonry two way walls
 1788:
 1789:
             last modified --
 1790:
 1791:
              7/03/91 (jpp)
             01/28/93 (dds) - added b to read, write; added debugging prints
 1792:
```

```
APPENDIX.B -29
| Saved: 4-28-93 2:37p
*----
1793:
                           modified ci to calculate density with gamma
1794:
1795:
           purpose --
1796:
            data is read from and/or written to a direct access file.
1797:
1798:
             in blast vulnerability guide, see section 5.0, figure 5.1*
1799:
1800:
            and section 6.1
1801:
1802:
            input --
1803:
            tapel5: direct access file with individual element data.
1804:
1805:
            output --
1806:
            tapel5 and arguments (see input discussion)
1807:
1808:
           restrictions
1809:
            none set in mau2wi
1810:
1811:
           call mdamrep ()
1812:
                       prints report on the most damaged components. the
                       report contains the information as follows:
1813:
                         most damaged building components report
1814:
1815:
                            wall/roof name
1816:
                            component type
1817:
                            percent damage
1818:
                            component local coordinates
1819:
                            p-i diagram terms, pbar
1820:
                            p-i diagram terms, ibar
1821:
           parameters: none
1822:
           1823:
           call onorm (xc, x1, x2, xb, vn)
1824:
1825:
            last modified --
1826:
            6/4/90 \text{ (map)}
1827:
1828:
           purpose --
1829:
            calculate outward normal vector from center of component
1830:
1831:
            method --
1832:
            see comments in code
1833:
1834:
            input --
1835:
            arguments:
1836:
             XC
                          coordinates of center of component
                          coordinates of two nodes on element which are not
1837:
             x1,x2
1838:
                          colinear with xc
                          coordinates of building orientation node
1839:
             хb
1840:
            output --
1841:
1842:
             arguments:
1843:
                          components of unit outward normal vector
              vn
1844:
1845:
            restrictions
1846:
            none set in onorm
1847:
1848:
           call optchk (option_table, key_pressed, match, old_key)
                       check if keypressed was option key or if mouse was
1849:
           function:
                        clicked within the range of the option field
1850:
1851:
                                option_table - table defining option area .
1852:
           parameters: inp
1853:
                                  where mouse can be clicked and what key
                                  that click translates to.
1854:
1855:
                                  1 form number
                                  2 row number of option
1856:
```

```
APPENDIX.B -30 |
| Saved: 4-28-93 2:37p
3 column number of option
1857:
                                   4 width of option field
1858:
                                   5 key which that area represents
1859:
                                   6 1: regular option
1860:
                                     2: external option field
1861:
1862:
                                key_pressed - key_pressed by user
                        inp
1863:
                                key_pressed - if was mouse click and within
1864:
                        out
                                   option field, key_pressed set to key
1865:
                                   represented by the click
1866:
                                match
1867:
                        out
                                   true : click or key pressed within option
1868:
1869:
                                          range
                                   false: click or key pressed not with
1870:
                                          option range
1871:
1872:
1873:
           call optionb (usrdat, key_pressed, form_number, ipos)
1874:
                       provides the options for forms in bdampost. however,
1875:
                       since no external options are used by the current
1876:
                       forms in bdampost, this is a dummy routine and only a
1877:
                       skeleton structure set up in case it might later be
1878:
                       needed.
1879:
1880:
           parameters:
                       usrdat
                                     : array containing form data
1881:
                                      : keystroke user left form with and
                       key_pressed
1882:
                                       is used as an input parameter into
1883:
                                       this routine to determine action to
1884:
                                       take
1885:
                                     : number of form being edited
1886:
                       form number
                                     : current row number in form (usrdat)
1887:
                       ipos
                                      when entered this routine
1888:
           1889:
           call optionb2 (usrdat, key_pressed, form_number, ipos) function: dummy program for editfrm2 since facedap forms have no
1890:
1891:
                       external options
1892:
1893:
           call out1(name, w, xchg, ninc, dxchg, xb, nbop, ndep, idep, nr)
1894:
1895:
1896:
            last modified --
1897:
              7/10/91 (map, jpp)
1898:
1899:
            purpose --
              writes input data to output file
1900:
1901:
1902:
            method --
1903:
1904:
1905:
            input --
1906:
1907:
1908:
            output --
1909:
1910:
1911:
            restrictions --
1912:
             none set in outl
              1913:
           call out2(w,xchg,dam,bd,br,bu,icount)
1914:
1915:
            last modified --
1916:
              7/10/91 (map, jpp)
1917:
1918:
1919:
            purpose --
              writes result to output file
1920:
```

```
*-----
! Saved: 4-28-93 2:37p
                                            APPENDIX.B -31
*----*
1921:
1922:
         method --
1923:
1924:
1925:
         input --
1926:
1927:
1928:
         output --
1929:
1930:
1931:
         restrictions --
1932:
          none set in out2
         -----
1933:
                             1934:
         call outpl1(name, w, ninc)
1935:
1936:
         last modified --
           7/30/91 (jpp)
1937:
1938:
1939:
         purpose --
1940:
           writes title, charge weight, and number of iterations to
1941:
           output plot file
1942:
1943:
         call outpl2(xb,xchg,bd,br,bu)
1944:
1945:
         last modified --
           7/30/91 (map, jpp)
1946:
1947:
1948:
         purpose --
1949:
          writes data to output plot file
1950:
                                    1951:
         dimension xb(3,5), xchg(3)
1952:
             sumr2 = 0.0
1953:
             do 100 i=1,3
1954:
             sumr2 = sumr2 + (xchg(i) - xb(i,1))**2
         100 continue
1955:
1956:
            r = sqrt(sumr2)
1957:
             write(3,*) r,bd,br,bu
1958:
            return
1959:
            end
1960:
                  1961:
         call outpost (dam, bd, br, bu)
1962:
1963:
         last modified --
1964:
          11/19/92 (dds) - created from out2
1965:
1966:
1967:
           writes result to output file for bdam postprocessor, bdampost
1968:
1969:
         method --
1970:
1971:
1972:
         input --
1973:
1974:
1975:
         output --
1976:
1977:
1978:
         restrictions --
1979:
          none set in outpost
1980:
         ______
1981:
         call pausl (msg)
1982:
        business systems integration
1983:
         (512) 680~3940
1984:
         copyright (c) 1988, 1989, 1990, 1991, 1992
```

```
APPENDIX.B
| Saved: 4-28-93 2:37p
all rights reserved
          call postprnt (damaged)
1987:
          function: provides a menu for user selection of postprocess
1988:
                      reports.
1989:
1990:
          parameters:
                                   : logical variable which indicates if
                      damaged
1991:
                                     building sustained any damage
1992:
                                     t : building sustained damage
1993:
                                     f : building sustained no significant
1994:
                                         damage (i.e. no component's damage
1995:
                                        exceeded 0%)
1996:
1997:
           call prephead (itype, report, line_kt)
1998:
           function: writes column headers for the preprocessor reports
1999:
                      based on itype and increments the line counter,
2000:
                       line_kt, based on number of lines written.
2001:
                                        type
                                                         # report
                          itype
2002:
                                        wall definitions
                                                           2
2003:
                           1
                           2
                                        component geometry
2004:
                                        frame geometry
                           3
2005:
                                        dependency
                           4
2006:
2007:
           parameters:
                       itype : type of header to be printed
2008:
                       report : indicates which report to print for the
2009:
                               specified itype (1 or 2)
2010:
                       line kt : current number of lines written to report file
2011:
                        2012:
           call prepprnt (stat_lun)
2013:
                       displays main print report menu to allow user to select
           function:
2014:
                       preprocessor reports to print. the following reports
2015:
                       are generated:
2016:
2017:
                         component properties
                         wall definitions
2018:
                         component geomtry
2019:
2020:
                         dependencies
           parameters:
2021:
                       stat_lun : logical unit connected to file which will
2022:
                                  contain the filenames of all reports files
2023:
                                  generated during a preprocessor session
2024:
             2025:
2026:
           call probrep ()
                      prints report containing proble description and
           function:
2027:
                       charge information
2028:
           parameters: none
2029:
            2030:
           call ptdepend (total_depend)
2031:
           function: saves the id numbers of the independent components
2032:
                       into the compg array, col. 14-17
2033:
           parameters:
2034:
                       total depend - total number of components in current
2035:
                         wall
2036:
2037:
           call ptgroup (wall_name, wall_index, itype, id,
2038:
                 set_gen_flag, w_comp_tmp)
2039:
                       if itype = 1, places the w_comp_table spreadsheet data
 2040:
           function:
                       for master and unique components for the specified
 2041:
                       wall/roof area into the compg storage area. if
 2042:
                       itype = 2, generated components from gencomp are placed
 2043:
                       in compg. if itype = 3, frame components are placed from
 2044:
                       frame_table in compg. compg contains all required
 2045:
                       information for each component used in the building.
 2046:
                       the rows within compg are never physically moved but
 2047:
                       order is maintained thru the link list mlink. pointers
 2048:
```

Saved:	4-28-93 2:37	p APPENDIX.B -33
2049:		to the beginning and end of the master/unique
2050:		components are maintained in head and tail. pointers to
2051:		the beginning and end of the generated nodes are
2052:		maintained in ghead and gtail. pointers to the beginning
2053:		and end of the frame components are maintained in fhead
054:	•	and ftail.
055:		
056:		compg contains the following for itypes 1 & 2:
057:		column 1 : wall name
058:		column 2 : component material type (wood, concrete, etc)
2059:		column 3 : component type (r/c beam, etc)
2060:		column 4 : component property name (user assigned)
2061:		column 5 : component id number
2062:		column 6 : x1 coordinates of end or corner
2063:		column 7 : yl coordinates of end or corner
2064:		column 8 : x2 coordinates of end or corner
2065:		column 9 : y2 coordinates of end or corner
2065:		column 10: master component indicator
2067:		yes - master
2068:		no - unique
2069:		column 11: center to center spacing
2070:		column 12: local direction
2071:		local x - generate in positive x direction
2072:		local y - generate in positive y direction
2073:		column 13: number of additional repeated components
2074:		
2075:		compg contains the following for itype 3:
2076:		column 1 : wall name - always frame
2077:		column 2 : component material type (concrete or steel)
2078:		column 3 : component type (r/c frame or steel frame)
2079:		column 4 : component property name (user assigned)
2080:		column 5 : component id number
2081:		column 6 : x1 coordinate of blastward exterior column,
2082:		coordinates are in the local coordinates of
2083:		the blastward wall
2084:		column 7 : y1 coordinate of blastward exterior column,
2085:		coordinates are in the local coordinates of
2086:		the blastward wall
2087:		column 8 : blank
2088:		column 9 : blank
2089:		column 10: set to no since no components may be generate
2090:		from this component
2091:		column 11: name of blastward wall
2092:		column 12: total frame height
2093:	parameters:	•
2094:	F	wall name - user assigned name of current wall/roof
2095:		area
2096:		wall index - index to wall geometry array
2090:		itype - indicates if storing master/unique,
2098:		generated or frame components
2098: 2099:		1 : master/unique component
2100:		2 : generated component
2100:		3 : frame components
2101:		id - id number of master component generated
2102: 2103:		valid only if itype = 2
		set gen flag - indicates if gen_flag is to be set
2104:		valid for generated components,
2105:		_ _
2106:		<pre>itype = 2 true - set gen_flag for master component</pre>
2107:		
2108:		used for generation to true
2109:		false - leave gen_flag alone
2110:		w comp tmp - temporary storage for the row containing
2111:		the master component that components are to be generated from. this is done since the work space,
2112:		

```
*_____
                                                    APPENDIX.B -34
| Saved: 4-28-93 2:37p
                          work table, is actually the same for master/unique
2113:
                         components and for the generated components.
2114:
2115:
2116:
          call ptmat (imat, icomp)
2117:
                      places the comp_table spreadsheet data for the
          function:
2118:
                      material property group specified by imat and
2119:
                      icomp from the xmp storage array. xmp contains
2120:
                      contains all required information for component
2121:
                      material property used in the building. the rows
2122:
                      within xmp are never physically moved but order is
2123:
                      maintained thru the link list clink.
2124:
2125:
                      xmp contains the following:
2126:
                      column 1 : icn - identification based on material
2127:
                                      and selected subset component
2128:
                      column 2 : component property name (user assigned)
2129:
                      column 3 : weighting factor
2130:
                      column 4 : beam width
2131:
                      column 5 : beam thickness
2132:
                      column 6 : loaded width
2133:
                      column 7 : total weight
2134:
                      column 8 : compressive concrete strength
2135:
                       column 9 : steel yield strength
2136:
                       column 10: depth to tensile steel
2137:
                       column 11: area tensile steel
2138:
                       column 12: moment of inertia
2139:
                       column 13: boundary condition
2140:
2141:
           parameters:
2142:
                                - material indicator
                       imat
2143:
                                 1 : concrete
2144:
                                  2 : steel
2145:
                                  3 : masonry
2146:
                                  4 : wood
2147:
                                - number of subcategory which occurs under
                       icomp
2148:
                                 each material
2149:
2150:
           call rclwi (ncl,nel)
2151:
2152:
           one way reinforced concrete slabs
2153:
2154:
2155:
           last modified --
            7/03/91 (jpp)
2156:
            01/27/93 (dds) - made moment of inertia be input, xim, rather than
2157:
                           calculated value, im and added debugging prints
2158:
2159:
2160:
             data is read from and/or written to a direct access file.
2161:
2162:
            method --
2163:
            see theory manual
2164:
2165:
            input --
2166:
            tapel5: direct access file with individual element data.
2167:
2168:
2169:
            tapel5 and arguments (see input discussion)
2170:
2171:
           restrictions
2172:
            none set in rclwi
2173:
            2174:
           call rc2wi(nc2,nel)
 2175:
 2176:
```

```
| Saved: 4-28-93 2:37p
                                                    APPENDIX.B -35 |
two way reinforced concrete slabs
2178:
2179:
           last modified --
2180:
            7/03/91 (jpp)
2181:
           01/27/93 (dds) - made moment of inertia be input, xim, rather than
2182:
                           calculated value, im; added iarch; and added debugging
2183:
                           prints
2184:
           purpose --
2185:
2186:
            data is read from and/or written to a direct access file.
2187:
2188:
          method --
2189:
           see theory manual
2190:
2191:
          input --
2192:
            tapel5: direct access file with individual element data.
2193:
2194:
           output --
2195:
            tapel5 and arguments (see input discussion)
2196:
2197:
           restrictions
2198:
           none set in rc2wi
2199:
                            _______
2200:
          call rcbmi (nc1, nel)
2201:
2202:
          reinforced concrete beams
2203:
2204:
           last modified --
2205:
            7/05/91 (jpp)
           11/18/92 (dds) - made moment of inertia be input, xim, rather than
2206:
2207:
                           calculated value, im
2208:
          01/27/93 (dds) - added debugging prints
2209:
2210:
          purpose --
2211:
            data is read from and/or written to a direct access file.
2212:
2213:
           method --
2214:
            see theory manual
2215:
2216:
           input --
2217:
            tapel5: direct access file with individual element data.
2218:
2219:
           output --
2220:
           tape15 and arguments (see input discussion)
2221:
2222:
          restrictions
2223:
           none set in rcbmi
2224:
                               ______
2225:
          call rceci (ncl,nel)
2226:
2227:
          exterior reinforced concrete columns
2228:
2229:
          last modified --
2230:
            7/05/91 (jpp)
2231:
           01/27/93 (dds) - made moment of inertia be input, xim, rather than
                          calculated value, im and added debugging prints
2232:
2233:
2234
          purpose --
2235:
            data is read from and/or written to a direct access file.
2236:
2237:
           method --
2238:
           see theory manual
2239:
2240:
           input --
```

```
APPENDIX.B +36
| Saved: 4-28-93 2:37p
.
*_____*
            tapel5: direct access file with individual element data.
2241:
2242:
           output --
2243:
            tapel5 and arguments (see input discussion)
2244:
2245:
            restrictions
2246:
            none set in rceci
2247:
2248:
           call rcici (ncl,nel)
2249:
2250:
            reinforced concrete interior columns
2251:
2252:
            last modified --
2253:
2254:
             7/05/91 (jpp)
            01/27/93 (dds) - made moment of inertia be input, xim, rather than
2255:
                            calculated value, im and added debugging prints
2256:
2257:
2258:
            purpose --
             data is read from and/or written to a direct access file.
2259:
2260:
2261:
            in blast vulnerability guide, see section 5.0, figure 5.1*
2262:
             and section 6.1
2263:
2264:
2265:
            input --
             tapel5: direct access file with individual element data.
2266:
2267:
            output --
2268:
            tapel5 and arguments (see input discussion)
2269:
2270:
2271:
           restrictions
            none set in rcici
2272:
2273:
           call rcmrfi (nr, nel)
2274:
2275:
            reinforced concrete frames
2276:
2277:
            last modified --
2278:
             7/12/91 (jpp)
2279:
            01/27/93 (dds) - made moment of inertia, xim and height of single story,
2280:
                             h, be inputs, rather than calculated values, im and h;
2281:
                             added debugging prints, added read for frame col. coord
2282:
2283:
2284:
            purpose --
              data is read from and/or written to a direct access file.
2285:
2286:
2287:
            method --
2288:
             see theory manual
2289:
2290:
             input --
             tapel5: direct access file with individual element data.
2291:
                         last record number in direct access file
              ireclast
2292:
2293:
2294:
              tapel5 and arguments (see input discussion)
2295:
2296:
2297:
             restrictions
             none set in romrfi
2298:
2299:
            call rcpsi (ncl,nel)
 2300:
 2301:
             prestressed concrete beams
 2302:
 2303:
             last modified --
 2304:
```

```
*-----
                                                        APPENDIX.B -37 |
| Saved: 4-28-93 2:37p
2305: 11/18/92 (dds) - created from rcbmi
2306:
           purpose --
2307:
            data is read from and/or written to a direct access file.
2308:
2309:
2310:
           method --
2311:
            in blast vulnerability guide, see section 5.0, figure 5.1*
2312:
            and section 6.1
2313:
2314:
           input --
            tapel5: direct access file with individual element data.
2315:
2316:
2317:
           output --
            tapel5 and arguments (see input discussion)
2318:
2319:
           restrictions
2320:
2321:
            none set in rcbmi
2322:
2323:
           call readat (name, w, xchg, ninc, dxchg, xb, nbop,
2324:
                                ndep,idep,nr,ireclast,iwrit)
2325:
2326:
           last modified --
2327:
              7/15/91
                        (jpp)
             11/19/92
                        (dds) - added xim to concrete beams read statement
2328:
2329:
             12/01/92
                        (dds) - commented out write to screen
                       (dds) - modified all required reads and write for all
2330:
            01/27/93
2331:
                               other component types
2332:
2333:
           purpose --
2334:
             reads input data and calls the proper subroutine
2335:
2336:
            method --
2337:
2338:
2339:
            input --
2340:
2341:
2342:
            output --
2343:
2344:
2345:
            restrictions --
2346:
            none set in readat
2347:
2348:
           call readf (formna, usrdat, noinp, ier, ipos, term_key, -
2349:
                 legalt, numtrm)
2350:
           business systems integration
2351:
           (512) 680-3940
           copyright (c) 1988, 1989, 1990, 1991, 1992
2352:
2353:
           all rights reserved
2354:
2355:
           call reflect( zlog, pres, imp )
2356:
           last modified --
2357:
            11/16/88 (tkb) initial release
2358:
2359:
2360:
            purpose --
              calculates reflected pressure and impulse
2361:
2362:
2363:
            method --
              uses fitted functions from arbr1-tr-02555 to calculate
2364:
2365:
              reflected pressure and impulse
2366:
2367:
            input --
2368:
              variable zlog in argument list
```

```
*-----
                                               APPENDIX.B -38
| Saved: 4-28-93 2:37p
          output --
2370:
           no files, just variables pres and imp in argument list
2371:
2372:
          restrictions --
2373:
2374:
           range of applicability:
2375:
2376:
               0.170 < z < 100.0 for z = 10.**zlog
2377:
                      2378:
          call reset ()
2379:
          function: resets required variables, forms, spreadsheets and
2380:
                     common blocks to 0s or blanks for beginning of new
2381:
                     problem.
2382:
          parameters:
2383:
                    none
2384:
          2385:
          call rightj (chosen_value, char_variable, variable_length)
          function: right justifies data within a variable. char_variable
2386:
                     contains the data to be right justified. chosen_value
2387:
2388:
                     contains the right justified data.
2389:
2390:
          call rsort2d (array, nval, isortcol, numrow, numcol, idir, -
2391:
               ier)
2392:
          business systems integration
2393:
          (512) 680-3940
2394:
          copyright (c) 1988, 1989, 1990, 1991, 1992
2395:
          all rights reserved
2396:
2397:
                     sort a real array
          function:
2398:
          parameters: array real array to sort
nval number of values to sort
2399:
2400:
                     isortcol column in array sort is based on
2401:
                     numrow number of rows to sort
                     numcol number of columns in array
2402:
2403:
                      idir sort direction (1 ascending, -1 descending)
2404:
                              0 : no errors occurred during sort
                      ier
2405:
                              1 : isortcol exceeded number of columns in
2406:
                                 arrav
2407:
2408:
           log = samecord(x, y, z)
2409:
           function: check if the 2 sets of coordinates specified in x, y,
2410:
                      and z are the same. the function returns a true if
2411:
                      if the 2 nodes have the same coordinates and a false
2412:
                      if the nodes are different
2413:
           parameters:
2414:
                             - array containing x coordinates for the 2
2415:
                               nodes
2416:
                              - array containing y coordinates for the 2
2417:
                               nodes
 2418:
                              - array containing z coordinates for the 2
 2419:
                              nodes
 2420:
                                       2421:
           call savedata (input_file_name)
 2422:
           function: saves data for current problem which is required
 2423:
                      for a later restart of the preprocessor.
 2424:
           parameters:
 2425:
                      input_file_name - name of file to save data to;
 2426:
                                       contains full path
 2427:
                2428:
           call setcid (wall_index, i, j, table, master)
 2429:
           function: assigns a unique component id for the specified
 2430:
                      component. the component id is generated in the
 2431:
                      following manner. the ten thousand and thousand
 2432:
```

```
| Saved: 4-28-93 2:37p
                                                     APPENDIX.B -39 |
position are reserved for the wall number, the
2433:
                       100s, 10s and 1s position are reserved for the
2434:
                       master or unique number a decimal point follows
2435:
2436:
                       and the decimal places refer to generated number
2437:
2438:
           parameters:
                      wall index - index to wall geometry array
2439:
                                - number of the master or unique component
2440:
                                - number of the generated component
2441:
2442:
                       table
                                - output variable which will hold the
                                  generated id number
2443:
2444:
                                 - true : indicates a master or unique is
                      master
2445:
                                    being generated so the decimal places
2446:
                                    will be 0
2447:
                                   false: slave is being generated, number
2448:
                                    will be assigned in the decimal places
2449:
           2450:
          call setfid (table)
                     assigns a unique frame id for the specified
2451:
           function:
                       frame. the frame id is generated in the
2452:
2453:
                       following manner. the ten thousand and thousand
                       position are 00 as opposed to regular components where
2454:
2455:
                      these positions are reserved for the wall number, the
                      100s, 10s and 1s position are reserved for the
2456:
                       frame number and the decimal places are 00.
2457:
2458:
2459:
          parameters:
                              - output variable which will hold the
                      table
2460:
                                  generated id number
2461:
2462:
                call sideon( zlog, pres, imp )
2463:
2464:
2465:
           last modified --
            11/16/88 (tkb) initial release
2466:
2467:
2468:
           purpose --
2469:
             calculates incident pressure and impulse
2470:
2471:
           method --
            uses fitted functions from arbrl-tr-02555 to calculate
2472;
2473:
             pressure and impulse
2474:
           input --
2475:
             variable zlog in argument list
2476:
2477:
2478:
           output --
             no files, just variables pres and imp in argument list
2479:
2480:
2481:
           restrictions --
2482:
2483:
             range of applicability:
2484:
                   0.170 < z < 100.0 for z = 10.**(zlog)
2485:
2486:
           call spreadop (chosen_value, option_key, col_position,
2487:
                 row position, col width, drow, dcol, spread no, edited field, term key, inbuf,
2488:
2489:
2490:
                 max rows)
                     provides the pop-up options on the spread sheet type
2491:
           function:
2492:
                      forms
2493:
           parameters:
                      chosen value - returns values selected by user or
2494:
                                   calculated by program
2495:
2496:
                       option_key - option key pressed on spreadsheet
```

```
------
                                                      APPENDIX.B -40 |
| Saved: 4-28-93 2:37p
col_position - current column on spreadsheet
2497:
                      row_position - current row on spreadsheet
                      col_width - width of current column
2499:
                                  - current row position in screen coordinates
                       drow
2500:
                                  - current col. position in screen coordinates
                       dcol
2501:
                       spread no - number of current spreadsheet
2502:
                      edited field - indicated if current field was edited
2503:
                                    0 - field not edited
2504:
                                    1 - field edited
2505:
                                   - last key user pressed during option
                       term_key
2506:
                                    selection
2507:
                                   - character array containing spreadsheet
                       inbuf
2508:
                                    values
2509:
                                   - maximum row in array used to store
                       max rows
2510:
                                    spreadsheet data, not necessarily the
2511:
                                     number of rows displayed on the
2512:
                                     spreadsheet. there could be less rows
2513:
                                    displayed
2514:
2515:
           call spreadop (chosen_value, option_key, col_position,
2516:
                 row position, col_width, drow, dcol, spread_no, edited_field, term_key, inbuf,
2517:
2518:
                 max rows)
2519:
                    provides the pop-up options on the spread sheet type
           function:
2520:
                       forms
2521:
2522:
           call stbmi (ncl,nel)
2523:
2524:
           steel beams
2525:
2526:
            last modified --
2527:
             7/09/91 (jpp)
2528:
            01/27/93 (dds) - added iten variable to read and write and added debugging
2529:
                            prints
2530:
2531:
            purpose --
2532:
            data is read from and/or written to a direct access file.
2533:
2534:
2535:
            method --
            see theory manual
2536:
2537:
            input --
2538:
             tape15: direct access file with individual element data.
2539:
2540:
            output --
2541:
            tape15 and arguments (see input discussion)
2542:
2543:
            restrictions
2544:
            none set in stbmi
2545:
             2546:
            call stcdi (ncl,nel)
2547:
2548:
            corrugated steel decking
2549:
2550:
            last modified --
2551:
             7/09/91 (jpp)
2552:
            01/27/93 (dds) - removed b from input variable and made a constant;
 2553:
                            added debugging prints
 2554:
 2555:
            purpose --
 2556:
             data is read from and/or written to a direct access file.
 2557:
 2558:
            method --
 2559:
             see theory manual
 2560:
```

```
| Saved: 4-28-93 2:37p
                                                     APPENDIX.B -41 |
    _______
2561:
           input --
2562:
            tapel5: direct access file with individual element data.
2563:
2564:
           output --
2565:
           tapel5 and arguments (see input discussion)
2566:
2567:
           restrictions
2568:
2569:
           none set in stcdi
                               _____
2570:
          call steci (ncl,nel)
2571:
2572:
2573:
           steel exterior columns
2574:
           last modified --
2575:
2576:
            7/10/91 (jpp)
           01/27/93 (dds) - added iten variable to read and write and added debugging
2577:
2578:
                           prints
2579:
          purpose --
2580:
            data is read from and/or written to a direct access file.
2581:
2582:
2583:
          method --
           see theory manual
2584:
2585:
           input --
2586:
           tapel5: direct access file with individual element data.
2587:
2588:
2589:
           tapel5 and arguments (see input discussion)
2590:
2591:
2592:
          restrictions
2593:
           none set in steci
2594:
2595:
          call stici (nc1, nel)
2596:
2597:
           steel interior columns
2598:
2599:
           last modified --
2600:
            7/11/91 (jpp)
           01/27/93 (dds) - added debugging debugging prints
2601:
2602:
          purpose --
2603:
            data is read from and/or written to a direct access file.
2604:
2605:
2606:
           method --
2607:
           in blast vulnerability guide, see section 5.0, figure 5.1*
2608:
            and section 6.1
2609:
2610:
          input --
           tapel5: direct access file with individual element data.
2611:
2612:
2613:
          output --
            tapel5 and arguments (see input discussion)
2614:
2615:
2616:
          restrictions
2617:
           none set in stici
2618:
2619:
          call stmrfi (nr, nel)
2620:
2621:
           steel frames
2622:
2623:
          last modified --
2624:
            7/12/91 (jpp)
```

```
APPENDIX.B -42 |
1 Saved: 4-28-93 2:37p
01/27/93 (dds) - made single story height, h, be input; removed idc
2625:
                           as input; removed calc of h; added debugging prints
2626:
                           added read for col.coord.
2627:
2628:
           purpose --
2629:
            data is read from and/or written to a direct access file.
2630:
2631:
2632:
           method --
           see theory manual
2633:
2634:
           input --
2635:
            tape15: direct access file with individual element data.
2636:
2637:
                         last record number in direct access file
             ireclast
2638:
2639:
2640:
            tape15 and arguments (see input discussion)
2641:
2642:
           restrictions
2643:
           none set in stmrfi
2644:
2645:
           call stmswi (ncl,nel)
2646:
2647:
           metal stud walls
2648:
2649:
            last modified --
2650:
             7/12/91 (jpp)
2651:
            01/27/93 (dds) - added debugging prints, all reference to ioptbc=2
2652:
                            removed
2653:
2654:
            purpose --
2655:
             data is read from and/or written to a direct access file.
2656:
2657:
2658:
            method --
            see theory manual
2659:
2660:
            input --
2661:
            tapel5: direct access file with individual element data.
2662:
2663:
            output --
2664:
            tapel5 and arguments (see input discussion)
2665:
2666:
            restrictions
2667:
            none set in stmswi
2668:
2669:
           call stowji (ncl,nel)
2670:
2671:
            open web steel joists
2672:
2673:
            last modified --
2674:
             7/08/91 (jpp)
2675:
            01/27/93 (dds) - removed variables from read and write which are no
2676:
                            longer required; added debugging prints, all
2677:
                            reference to web shear failure deleted
2678:
2679:
2680:
             data is read from and/or written to a direct access file.
2681:
2682:
            method --
2683:
             see theory manual
2684:
 2685:
2686:
            input --
             tapel5: direct access file with individual element data.
 2687:
 2688:
```

```
APPENDIX.B -43
| Saved: 4-28-93 2:37p
*----
         output --
2690:
           tapel5 and arguments (see input discussion)
2691:
          restrictions
2692:
2693:
           none set in stowji
          ______
2694:
2695:
          call sum(dam,bd,br,bu)
2696:
          last modified --
2697:
2698:
            7/10/91 (map, jpp)
2699:
2700:
           purpose --
2701:
            summation of building damage
2702:
2703:
          method --
2704:
2705:
2706:
           input --
2707:
2708:
2709:
           output --
2710:
2711:
          restrictions --
2712:
           none set in sum
2713:
            _______
2714:
          call threat (r, w, iopt2)
2715:
2716:
           last modified --
2717:
             8/16/90 (mag)
2718:
2719:
           purpose --
2720:
2721:
             definition of charge weight and standoff from common building
2722:
           method --
2723:
             charge weight and standoff are directly input
2724:
2725:
2726:
           input --
2727:
             none
2728:
2729:
           output --
2730:
             charge weight and standoff
2731:
2732:
           restrictions
            charge weight must be a value between 35.0 and 4000.0 pounds
2733:
2734:
          call updatgen (wall_name, wall_index, ier)
2735:
                     checks all master/unique components for the specified
2736:
          function:
                     wall to determine if any master components have not
2737:
                     been generated. if a master is found whose components
2738:
                     have not been generated, the components will be generated
2739:
2740:
                     and the generated flag updated.
2741:
          parameters:
2742:
                     wall name - name of current wall/roof area
                     wall index - index into the wall area used to
2743:
2744:
                               access wall number for current wall name
                               - 0 : no error occurred
2745:
                     ier
                               1 : one or more required pieces of data
2746:
                                   in missing from the current master
2747:
2748:
          call validchg (lun, fatal_error, error_occurred,
2749:
               total_fatal, total_warnings)
2750:
2751:
                     check the position of the charge to determine if the
          function:
                     charge's location meets minimum requirements of program
2752:
```

*			
Saved:	4-28-93	2:37p	APPENDIX.B -44
2753: 2754: 2755: 2756: 2757:			these are that the scaled standoff is greater than 1.0 ft/lb^(1/3) and that charge cannot be inside building or over roof. also issues warnings when charge is within scaled standoff of 3.0 ft/lb^(1/3) or when charge height
_	-		this may invalidate surface burst assumption in blast
2758:			loads calculations.
2759:	paramet		
2760:	parame		lun : logical unit number of file messages
2761:			are written to if to screen if false
2762:			fatal_error : logical variable indicating if fatal
2763: 2764:			error occurred. this variable is
2765:			initialized once at the beginning of
2766:			the validation process to false and set
2767:			to true each time a fatal error occurs
2768:			f - no error occurred
2769:			t - fatal error occurred
2770:			error_occurred : logical variable indicating if a
2771:			fatal error occurred in the checking
2772:			done by this subroutine
2773:			total fatal : total number of fatal errors
2774:			total warnings: total number of warning errors
2775:			
2776:	call	valido	mp (lun, fatal_error, error_occurred,
2777:			fatal) performs validation checking of component properties.
2778:	functi	on:	a component has an underlined value.
2779:			a component had a value less than or equal to v.
2780:			3. components have unique name for each category.
2781:			
2782:	parame	ters.	lun - logical unit number for error output
2783:			eile.
2784: 2785:			a logical variable indicating if fatal
2785: 2786:			courted this variable is initialized once
2787:			the haginning of the validation process to laise
2788:			and set to true each time a fatal error occurs
2789:			f - no error occurred
2790:			t - fatal error occurred
2791:			error_occurred - indicates error occurred in this
2792:			category
2793:			total_fatal - total number of fatal errors
2794:			total latal - total manor of terminate total later
2795:			ond (lun, fatal_error, error_occurred,
2796:	CSII	ASTIG	faral formings:
2797:	functi		performs validation checking of dependency data. The
2798:	Lunce		Eallowing checks are made:
2799: 2800:			1 total number of dependency pairs are checked to
2801:			see if they exceed the maximum allowable number
2802:			of pairs (fatal)
2803:			2. each wall/roof area is checked to see if the
2804:			dependencies for that area have been generated.
2805:			(warning)
2806:			3. each row of compg is checked to see make certain that all independent ids in the row are unique
2807:	•		within that row. (fatal)
2808:			A anch independent id is checked to make sure that
2809:			it has a corresponding dependent id within compg.
2810:			(fatal)
2811:		a+ c = - ·	•
2812:	param	eters:	lun : logical unit number of file messages
2813:			are written to if to screen if talse
2814: 2815:			sala arror · logical variable indicating if fatal
2815: 2816:			error occurred. this variable is
2010.			

*	:
Saved:	4-28-93 2:37p APPENDIX.B -45
2817:	
2818:	initialized once at the beginning of
	the validation process to false and set
2819:	to true each time a fatal error occurs
2820:	f - no error occurred
2821:	t - fatal error occurred
2822:	error_occurred : logical variable indicating if a
2823:	fatal error occurred in the checking
2824:	done by this subroutine
2825:	total_fatal : total number of fatal errors
2826:	total_warnings : total number of warning errors
2827:	
2828:	call validwal (lun, fatal error, error occurred, -
2829:	total fatal, total warnings)
2830:	function: performs validation checking of wall data.
2831:	 no walls may have identical coordinates
2832:	2. wall names must be unique and no wall/roof may
2833:	be named frame.
2834:	3. no blank coordinates
2835:	4. coordinates in a wall must be planar
2836:	5. coordinates were entered in clockwise or
2837:	
	counterclockwise order
2838:	6. y coordinates must be non-negative7. 3rd coordinate must > 4th coordinate
2839:	
2840:	8. walls must connect
2841:	parameters:
2842:	lun : logical unit number for error output
2843:	file
2844:	<pre>fatal_error : logical variable indicating if fatal</pre>
2845:	error occurred. this variable is
2846:	initialized once at the beginning of
2847:	the validation process to false and set
2848:	to true each time a fatal error occurs
2849:	f - no error occurred
2850:	t - fatal error occurred
2851:	error occurred : indicates error occurred in this
2852:	category
2853:	total fatal : total number of fatal errors
2854:	total warnings : total number of warning errors
2855:	
2856:	call validwcmp (lun, fatal error, error occurred, -
2857:	total fatal, total warnings)
2858:	function: performs validation checking of wall component
2859:	definition data.
2860:	1. check if component type is defined
2861:	
2862:	3. check if mismatch occurs between material type
2863:	and component type
2864:	4. check if mismatch occurs between component type
2865:	and component property name
2866:	5. check if component property name for each item in
2867:	compg is in xmp table
2868:	6. check for component specific type errors (done
2869:	by itypechk which is called by chkcomp)
2870:	6. coordinates for a given wall must be unique
2871:	7. coordinates of component must be in bounds of
2872:	wall.
2873:	8. walls must connect
2874:	9. coordinates must define legitimate beam or plate
2875:	parameters:
2876:	lun : logical unit number of file messages
2877:	are written to if to screen if false
2878:	fatal error : logical variable indicating if fatal
2879:	error occurred. this variable is
2880:	initialized once at the beginning of
2000.	and the second s

```
APPENDIX.B -46
| Saved: 4-28-93 2:37p
*-----*
                                         the validation process to false and set
2881:
                                          to true each time a fatal error occurs
2882:
                                            f - no error occurred
2883:
                                            t - fatal error occurred
2884:
                         error occurred : logical variable indicating if a
2885:
                                          error occurred in the checking
2886:
                                           done by this subroutine
2887:
                                       : total number of fatal errors
                         total fatal
2888:
                        total_warnings : total number of warning errors
2889:
            2890:
            int = wallchk (wall index)
2891:
            function: to check if four nodes in wall area in row "wall index"
2892:
                         are coplanar and if four nodes are entered either
2893:
                         clockwise or counterclockwise and print error message
2894:
                         detailing any error that is located
2895:
                         if there are no problems the wallchk = 0
2896:
                         if there are problems, wallchk = 1
2897:
            parameters:
2898:
                         wall index - index into the wall area used to
2899:
                            access wall number for current wall_name
2900:
                           2901:
            call walldef (ier)
2902:
                        allows user input of the wall/roof areas. user
            function:
2903:
                         inputs a 10 character unique name for the area
2904:
                          followed by the global x, y and z coorindates for
2905:
                         the four corner points. a maximum of 50 unique wall/roof
2906:
                          areas are allowed. this maximum is set in sprdhead.dat.
2907:
                         the dimensioning of wall table allows for up to
2908:
                         max row ever columns. the following checks are
2909:
                         performed by walldef:
2910:
                            1. checks if user changed name of a wall/roof area
2911:
                                so that compg can be updated. the old name
2912:
                                becomes the new name in compg.
2913:
                            2. checks for a deleted row in the wall/roof table
2914:
                                so that compg is kept current. all references
2915:
                                to that wall/roof area are deleted from compg.
2916:
                            3. checks that all wall/roof areas have unique names.
2917:
                            4. checks that no wall/roof area has the name frame,
2918:
                                since this is a reserved word.
2919:
                            5. checks if 2 wall/roof areas have identical
2920:
                                coordinates.
2921:
                            checks if a wall/roof area has 2 identical nodes.
2922:

    checks for overlapping or gaps in wall/roof areas

2923:
2924:
                          wall table contains the following data in each column:
2925:
2926:
                           1. user defined unique name (user entry)
2927:
                           2. global x coordinate for 1st corner point (user entry)
2928:
                           3. global y coordinate for 1st corner point (user entry)
2929:
                           4. global z coordinate for 1st corner point (user entry)
                           5. global x coordinate for 2nd corner point (user entry)
2930:
2931:
                              global y coordinate for 2nd corner point (user entry)
                           6.
2932:
                           7. global z coordinate for 2nd corner point (user entry) 8. global x coordinate for 3rd corner point (user entry)
2933:
2934:
                           9. global y coordinate for 3rd corner point (user entry)
10. global z coordinate for 3rd corner point (user entry)
11. global x coordinate for 4th corner point (user entry)
12. global y coordinate for 4th corner point (user entry)
2935:
2936:
2937:
2938:
                           13. global z coordinate for 4th corner point (user entry)
2939:
                           14. wall id number (program use)
2940:
                           15. characters 1-3 number of components for wall
2941:
                                area (program use)
 2942:
                               characters 8-9 used for roof connectivity in
 2943:
                                 bldgchk (program use)
```

2944:

```
| Saved: 4-28-93 2:37p
                                                      APPENDIX.B -47 |
character 10 used to indicate if dependencies have
2945:
2946:
                               been generated:
                                 t - dependencies have been generated for
2947:
2948:
                                     current wall/roof area
                                 f - dependencies have not been generated for
2949:
                                     current wall/roof area
2950:
          parameters:
                                      - 0 : no errors occurred
2952:
                         ier
                                        1 : duplicate wall/roof names
2953:
                                        2 : 2 wall/roof areas have identical
2954:
2955:
                                           coordinates
                                        3 : a wall/roof area has 2 identical
2956:
2957:
                                           nodes
                                        4 : wall/roof area not coplanar
2958:
2959:
                                        5 : overlap or gap in wall/roof area(s)
                                       6 : problem in connectivity
2960:
           2961:
           log =n wallok (lun, to_screen, fatal_error,
2962:
                  error_occurred, total_fatal, total_warnings)
on: checks if the wall/roof areas match together at corners
2963:
2964:
           function:
2965:
                        to form a building without gaps or overlapping wall
                        areas. if a problem is found, the function returns
2966:
                        false after printing out an error message. a true
2967:
                        issued by this subroutine does not guarentee that the
2968:
2969:
                        building is well defined, but a false is indicative
2970:
                        that there is a problem.
2971:
           parameters:
                                       : logical unit number of file messages
2972:
                        lun
                                        are written to if to_screen if false
2973:
                                       : logical variable indicates where
2974:
                        to_screen
2975:
                                        error messages will be displayed
                                        t : display to screen
2976:
                                        f : send to file indicated by logical
2977:
2978:
                                            unit lun
2979:
                        fatal error
                                       : logical variable indicating if fatal
                                        error occurred. this variable is
2980:
2981:
                                         initialized once at the beginning of
2982:
                                        the validation process to false and set
2983:
                                         to true each time a fatal error occurs
2984:
                                          f - no error occurred
                                          t - fatal error occurred
2985:
2986:
                        error occurred : logical variable indicating if a
                                         fatal error occurred in the checking
2987:
2988:
                                        done by this subroutine
                        total fatal : total number of fatal errors
2989:
                        total warnings: total number of warning errors
2990:
2991:
2992:
           call wallrep ()
2993:
           function:
                        generates the wall definition's report sorted by the
                        wall name. two reports are required to provide all
2994:
2995:
                        information. the first report prints the wall/roof
                        name with coordinates of corner 1 and corner 2. the
2996:
                        second report prints the wall/roof name with
2997:
                        coordinates of corner 3 and corner 4.
2998:
2999:
           parameters:
3000:
                       none
3001:
           call wdamrep (wall name)
3002:
                       prints report on damaged components for specified wall.
3003:
           function:
3004:
                        one of two reports is generated, depending on if the
                        it is a standard wall/roof area or a frame. the report
3005:
3006:
                        contains the following information:
3007:
                         for non-frame components:
3008:
                             damaged building components report -
```

```
APPENDIX.B -48
| Saved: 4-28-93 2:37p
                                component type
3009:
                               percent damage
3010:
                                component local coordinates
3011:
                               repair or replace
3012:
                                p-i diagram terms, pbar
3013:
                               p-i diagram terms, ibar
3014:
                          for frame components:
3015:
                             damaged building components report -
3016:
                                component type
3017:
                                percent damage
3018:
                                blastward wall name
3019:
                                component local coordinates
3020:
                                repair or replace
3021:
                                p-i diagram terms, pbar
3022:
                                p-i diagram terms, ibar
3023:
3024:
           parameters:
                                     : user assigned name of wall/roof area,
                        wall name
3025:
                                        unless the area is a frame in which
3026:
                                        case the program assigned the area name
3027:
                                        of frame
3028:
3029:
           call wdbmi (ncl,nel)
3030:
3031:
            wood beams
3032:
3033:
            last modified --
3034:
             7/08/91 (jpp)
3035:
             01/28/93 (dds) - made moment of inertia be input, xim, rather than
3036:
                             calculated value, im; added sigy; added debugging
3037:
                             prints;
3038:
                             modified ci and cp
3039:
3040:
             purpose --
3041:
             data is read from and/or written to a direct access file.
3042:
3043:
             method --
3044:
             see theory manual
3045:
3046:
             input --
3047:
             tapel5: direct access file with individual element data.
3048:
3049:
3050:
             tape15 and arguments (see input discussion)
3051:
3052:
            restrictions
3053:
             none set in wdbmi
3054:
3055:
            call wdeci (ncl,nel)
3056:
3057:
             exterior wood columns
3058:
3059:
             last modified --
3060:
              7/08/91 (jpp)
3061:
             01/27/93 (dds) - made moment of inertia be input, xim, rather than
 3062:
                              calculated value, xim; added sigy; added debugging print:
 3063:
                              modified ci and cp
 3064:
 3065:
             purpose --
 3066:
             data is read from and/or written to a direct access file.
 3067:
 3068:
             method --
 3069:
             see theory manual
 3070:
 3071:
 3072:
             input --
```

```
| Saved: 4-28-93 2:37p
                                                   APPENDIX.B -49 |
tape15: direct access file with individual element data.
3073:
3074:
          output --
3075:
           tapel5 and arguments (see input discussion)
3076:
3077:
          restrictions
3078:
           none set in wdeci
3079:
3080:
3081:
          call wdici (ncl,nel)
3082:
3083:
          interior wood columns
3084:
3085:
           last modified --
3086:
            7/09/91 (jpp)
           01/28/93 (dds) - made moment of inertia be input, im, rather than
3087:
3088:
                          calculated value; added debugging prints
3089:
3090:
          purpose ---
           data is read from and/or written to a direct access file.
3091:
3092:
          method --
3093:
3094:
           see theory manual
3095:
3096:
           input --
           tape15: direct access file with individual element data.
3097:
3098:
           output --
3099:
3100:
           tape15 and arguments (see input discussion)
3101:
3102:
          restrictions
3103:
           none set in wdici
                              3104:
3105:
         call wdrfi (ncl,nel)
3106:
3107:
          wood roofs
3108:
3109:
          last modified --
3110:
            7/11/91 (jpp)
          01/28/93 (dds) - added sigy; added debugging prints
3111:
                         modified ci and cp
3112:
3113:
          purpose --
3114:
3115:
           data is read from and/or written to a direct access file.
3116:
          method --
3117:
3118:
           see theory manual
3119:
3120:
           input --
           tapel5: direct access file with individual element data.
3121:
3122:
3123:
           tapel5 and arguments (see input discussion)
3124:
3125:
3126:
          restrictions
3127:
           none set in wdrfi
                            _______
3128:
          call wdwli (nc1,nel)
3129:
3130:
3131:
           wood walls
3132:
3133:
           last modified --
            7/11/91 (jpp)
3134:
           01/28/93 (dds) - added sigy to read and write and added debugging
3135:
3136:
                          prints
```

```
APPENDIX.B -50
| Saved: 4-28-93 2:37p
3137:
                           modified ci and cp
3138:
            purpose --
3139:
            data is read from and/or written to a direct access file.
3140:
3141:
           method --
3142:
            see theory manual
3143:
3144:
            input --
3145:
            tapel5: direct access file with individual element data.
3146:
3147:
3148:
            tapel5 and arguments (see input discussion)
3149:
3150:
           restrictions
3151:
            none set in wdwli
3152:
3153:
           int =wnindex (comp_id)
3154:
           function: strips the wall area number off the component id number
3155:
                       and searches wall table until it finds the row with
3156:
                       this wall area number in column 14. the function name
3157:
                       wnindex returns the row number. if the id number is
3158:
                       for a frame component (00 are first 2 digits), the
3159:
                       index into wall_table for the blastward wall is returned
3160:
                       in the function name
3161:
           parmaeters :
3162:
                                         : component id which needs a
                       comp id
3163:
                                          corresponding wall index
3164:
           ______
3165:
           call wrthead (lun, itype, line_kt)
3166:
                      writes column headers based on itype and increments the
3167:
           function:
                        line counter, line_kt, based on number of lines written.
3168:
                                         subroutine report
                          itype
3169:
                                                         1
                                            mbamrep
                            1
3170:
                                                         1 non-frame
                            2
                                            wdamrep
3171:
                                                         1 frame
                                            wdamrep
                            3
3172:
                                                         1 non-frame
                                            blstrep
                            4
3173:
                                                         1 frame
                            5
                                            blstrep
3174:
           parameters:
3175:
                        lun : logical unit connected to report file
itype : type of header to be printed
3176:
3177:
                       line kt : current number of lines written to report file
3178:
3179:
           call xylimit (to_screen, lun, wall_index, xmax, ymax,
3180:
                  xmin, ymin, ier)
3181:
3182:
                        to calculate the maximum and minimum local coordinates
           function:
3183:
                        in the wall area in the row of wall_table indentified
3184:
                        by the pointer wall_index
3185:
           parameters:
3186:
                                      - logical variable indicates where
3187:
                        to_screen
                                        error messages will be displayed
3188:
                                        t : display to screen
3189:
                                        f : send to file indicated by logical
3190:
                                           unit lun
3191:
                                      - logical unit number for error output
                        lun
3192:
                                       file
3193:
                                      - index into the wall area used to
                        wall index
3194:
                                       access wall number for current
3195:
                                       wall name
3196:
                                      - maximum x value for specified wall
                        xmax
3197:
                                     - maximum y value for specified wall
3198:
                        ymax
                                     - minimum x value for specified wall
                        \mathbf{z}min
3199:
                                     - minimum y value for specified wall
                        ymin
3200:
```

Saved:	4-28-93 2:	37p	APPENDIX.B -51
3201:		ier	- error flag
3202:			0 : no errors occurred
3203:			1 : all coordinates for current wall
3204:			were 0
3205:			2 : error occurred converting wall
3206:	•		coordinates from character to
3207:			real
3208:			3 : wall name not defined
3209:			
3210:	call xyz2	(windex, j,	xyz, x)
3211:	function:		al x,y,z coordinates of j local coordinates
3212:			x and places them in vector xyz. the local
3213:			s in x are in the wall area designated by
3214:			inter windex in wall table. each point in
3215:			coordinates. each point in xyz has 3*j
3216:		coordinate	
3217:	parameters	:	
3218:	•		ndex into the wall area used to access wall
3219:		n	umber for current wall name
3220:			umber of coordinates sent; should be no
3221:			ore than 4 pairs of coordinates
3222:			lobal coordinates
3223:			ocal coordinates
3224:			
3225:	call xyzc	oord (mindex,	xyz, id)
3226:	function:		al x,y,z coordinates of two component endpoi
3227:			points for component in row 'mindex' in
3228:			ix if mindex > 0
3229:		if mindex=), then the the variable 'id', which should
3230:			e the component id number, is used to determ
3231:		mindex	
3232:		(xyz(i), i=	1,3) global coordinates of first point
3233:			4,6) global coordinates of second point
3234:	parameters	_	•
3235:	=	mindex :	flag indicating to use component id or row {
3236:			0 - flag to use component id to find desir
3237:			component in compg and place component
3238:			endpoints in xyz
3239:			>0 - row in compg for coordinates to be
3240:			placed in xyz
3241:		xyz :	global coordinates of the two endpoints
3242:			of the desired component
3243:			id number of desired component, only used
3244:			if mindex = 0

.

•

Appendix C

Description of Primary Subroutines Called in the FACEDAP Code that are Unique to Single Component Analysis

```
| Saved: 5-16-94 5:38p
                                                         APPENDIX.C -1
 _____
           call bloadc ()
           function:
                        This routine is designed for future expansion when
  3:
  4:
                        the user is allowed to enter the charge information
  5:
                        by charge weight, standoff and free-field or reflected
  6:
                        or by blast pressure and impulse
  7:
           parameters: none
  8:
           call comprep ()
  9:
 10:
           function:
                        Prints report on component damage for Single Component
 11:
                        Analysis
 12:
           parameters: none
 13:
 14:
           call cprphead (lun, itype, report, line kt)
 15:
                      Writes column headers for the COMPONENT Preprocessor
           function:
 16:
                        Component Property reports based on itype and report and
 17:
                        increments the line counter, line_kt, based on number
 18:
                        of lines written.
 19:
                           itype
                                            property
                                                           # reports
                                             RCBMI
 20:
                             1
                                                               2
 21:
                             2
                                             RC1WI
                                                               2
 22:
                                             RC2WI
 23:
                                                               2
                             4
                                             RCECI
 24:
                             5
                                                               2
                                             RCICI
 25:
                             6
                                                               2
                                             RCMRFI
 26:
                             7
                                                               2
                                             RCPSI
 27:
                             8
                                             STBMI
 28:
                             9
                                                               1
                                             STMSWI
 29:
                            10
                                                               1
                                             STOWJI
 30:
                            11
                                             STCDI
                                                               2
 31:
                            12
                                             STECI
 32:
                            13
                                             STICI
                                                               2
 33:
                            14
                                             STMRFI
                                                               2
 34:
                            15
                                             MAU1WI
 35:
                            16
                                             MAU2WI
                                                               2
 36:
                            17
                                             MAR1WI
 37:
                                                               2
                            18
                                             MAR2WI
 38:
                            19
                                             MAPILI
 39:
                            20
                                             WDWLI
 40:
                            21
                                             WDRFI
 41:
                            22
                                             WDBMI
 42:
                            23
                                             WDECI
 43:
                                             WDICI
                            24
 44:
           parameters:
 45:
                               : logical unit connected to report file
                        lun
 46:
                                : type of header to be printed
                        itype
 47:
                        report : indicates which report to print for the
 48:
                                  specified itype (1 or 2)
 49:
                        line_kt : current number of lines written to report file
 50:
           call filxist2 (input_file_name, ier)
 51:
 52:
           routine
                     : FILXIST2
 53:
           function
                      : Allows user to select the COMPPREP file name to save
 54:
                        the file under. The default is the current file name.
 55:
                        Issues warning message if the selected file already
 56:
                        exists. User then has the choice of writing over the
 57:
                        existing file or aborting the save.
 58:
           parameters :
                                           : Current Preprocessor file name
 59:
                        input file name
 60:
                                            selected by user. Contains full
 61:
                                            path
                                           : 0 - Save file
 62:
                        ier
                                           : 1 - Do not save file
 63:
 64:
```

```
APPENDIX.C -2
| Saved: 5-16-94 5:38p
call getcomp (input_file_name)
         function: Retrieve data for current problem which is required
 66:
                      for a restart of the COMPONENT Preprocessor, COMPPREP.
 67 .
         parameters:
 68:
                      input_file_name : name of file to retrive data
 69:
                                       from; contains full path
 70:
          -
 71:
          call getdefct (mat name, menu, num_comps)
 72:
          function: Provides a pop-up menu for COMPPREP of all component
 73:
                      types which has properties defined for them. This is
 74:
                      done by checking the XMP array.
 75:
          parameters: mat name - material name of selected material
 76:
                      menu - array to store component types that have been used
 77:
 78:
                      num_comps - total number of component types that have been used
 79:
 80:
              81:
          call getdefmt (menu, num mats)
 82:
          function: Provides a pop-up menu for COMPPREP of all materials
 83:
                      which has properties defined for them. This is done
 84:
                      by checking the XMP array.
 85:
          parameters: menu - array to store material names that have been used
 86:
 87:
                      num mats - total number of materials that have been used
  88:
  89:
          call getspred (spread no, ier)
  90:
                     Reads all required variables for the specified
 91:
           function:
                      spreadsheets from the data file SPRDHED2.DAT.
  92:
                      This routine is used by COMPPREP.
  93:
                      The following values are read:
  94:
                      1. Spread no
  95:
                      2. Number of columns on spreadsheet
 96:
                      3. Number of rows on spreadsheet
  97:
                      4. Number of column header lines
 98:
                      5. Column Headers
 99:
                      6. Help text
 100:
                      7. Variable type for each column (i.e. character,
 101:
                            integer, real)
 102:
                      8. Number of defaults
 103:
                      9. Column numbers of columns which have default settings
 104:
                      10. Default value for the default columns
 105:
                      11. Fields where duplication on spreadsheet allowed
 106:
                         0 - field can be duplicated
 107:
                         1 - field can not be duplicated
 108:
           parameters:
 109:
                                - number of spreadsheet to retrieve data for
                      spread no
 110:
                                - 0 : no error occurred
 111:
                      ier
                                  1 : error occurred opening file,
 112:
                                     SPRDHEAD.DAT
 113:
                                  2 : error occurred reading SPRDHED2.DAT
 114:
 115:
           call getvars (spread_no, col_position, calc_type)
 116:
           function: Used by spreadop for determining the columns used in
 117:
                       default calculations. Based on the spreadsheet number,
 118:
                       spread no and the column position on the spreadsheet,
 119:
                       col position, the required column numbers are returned
 120:
                       along with an indicator of the equation to be used for
 121:
                       calcluating. This routine is used for the Component
 122:
                      Analysis, COMPPREP.
 123:
           124:
           call gtshort (comp_type_name, icomp, imat, ier)
 125:
                      Uses the specified component property name to find
           function:
 126:
                       the icomp and imat that corresponds to the specified
 127:
                       type. Comp_type_name is the full name.
 128:
```

*						
Saved:	5-16-94 5:38p APPENDIX.C -3					
129:	parameters:					
130:	comp type name : name of component type					
131:	icomp : component # with a given material group					
132:	imat : material #					
133:	ier : 0 - no error occurred					
134:	1 - component type name not found					
135:						
136:	call initcomp ()					
137:	function: Initializes the COMPDEF.FRM with the first component					
138:	it finds in XMP.					
139:	parameters: none					
140:						
141:	call inpfil2 (file_name, pattern, ier)					
142:	function: This routine was modified from the IOSUB INPFIL routine.					
143:	This routine has been specifically modified to strip off					
144:	the .BLG extension. Calling with any other extension					
145:	will require modification of this code					
146: 147:	parameters: file_name file to search for					
147:	pattern input pattern					
149:	ier error code, 0=normal, 1=error					
150:	call optionb (usrdat, key pressed, form_number, ipos)					
151:	function: Provides the options for forms in BDAMPREP and COMPPREP.					
152:	parameters:					
153:	usrdat : array containing form data					
154:	key pressed : keystroke user left form with and					
155:	is used as an input parameter into					
156:	this routine to determine action to					
157:	take					
158:	form_number : number of form being edited					
159:	ipos : current row number in form (usrdat)					
160:	when entered this routine					
161:						
162: 163:	call probrep2 ()					
163: 164:	function: Prints report containing problem description and					
165:	load information for the Single Component Preprocessor					
166:	parameters: none					
167:	call resetc ()					
168:	function: Resets required variables, forms, spreadsheets and					
169:	common blocks to 0s or blanks for beginning of new					
170:	problem for the Component Preprocessor, COMPPREP.					
171:	parameters:					
172:	none					
173:	# # # # # # # # # # # # # # # # # # #					
174:	call savecomp (input_file_name)					
175:	function: Saves data for current problem which is required					
176:	for a later restart of the COMPONENT Preprocessor,					
177:	COMPPREP.					
178:	parameters:					
179: 180:	<pre>input_file_name - name of file to save data to;</pre>					
100:	Contains full path					